공학석사학위논문

평면의 임의 대응쌍을 이용한 반복적 3차원 정합 기법

An Iterative 3D Registration Algorithm using Random Pair of Plane Patches

2015년 8월

서울대학교 대학원

전기 컴퓨터 공학부

문 지 훈

공학석사학위논문

평면의 임의 대응쌍을 이용한 반복적 3차원 정합 기법

An Iterative 3D Registration Algorithm using Random Pair of Plane Patches

2015년 8월

서울대학교 대학원

전기 컴퓨터 공학부

문 지 훈

평면의 임의 대응쌍을 이용한 반복적 3차원 정합 기법

An Iterative 3D Registration Algorithm using Random Pair of Plane Patches

지도교수 이 범 희

이 논문을 공학석사 학위논문으로 제출함

2015 년 8 월

서울대학교 대학원

전기 컴퓨터 공학부

문 지 훈

문지훈의 공학석사 학위논문을 인준함

2015 년 8 월

위 원 장 _____

부위원장 _____

위 원 _____

초록

본 논문은 3차원 공간에서 평면 패치를 기반으로 이미지 정합을 하는 새로운 기법을 제시한다. 정합에 사용되는 대부분의 알고리즘은 특정점 간의 유사도를 이용하여 대응관계를 구하고 이를 기반으로 두 좌표계 사이의 강체변환을 구한다. 그러나 대응관계를 구하는 문제는 특징점에 대한 정보가 부족하거나 이상점이 발생하는 경우 부정확한 결과를 초래할 수 있고 이는 곧 정합의 실패에 영향을 미칠 수 있다.

본 논문에서는 이러한 문제를 해결하기 위하여 평면 패치들의 집합으로 이루어진 두 3차원 좌표계에서 각각 임의의 평면을 추출한 후 거리 제곱 평균 함수의 값을 계산하여 두 좌표계 간의 유사도를 측정한다. 이 과정을 반복적으로 수행하여 정합하고자 하는 프레임을 가장 유사하게 만드는 강체변환을 결정한다. 그 다음 고정된 강체변환에 대하여 거리 제곱 평균 함수의 값을 최소화 시키는 방향으로 평행이동 벡터를 보정하여 정합을 완료한다.

본 논문의 기법은 대응관계를 찾는 데 걸리는 시간을 줄일 수 있고 이상점에 강인하다는 데 의의가 있으며, 이를 시뮬레이션 및 실제 환경에서의 실험을 통해 검증하였다.

주요어: 평면 패치, 정합, 임의 대응쌍, 거리 제곱 평균

학번: 2013-23114

목차

초록 **i**

제 1 장 Introduction **1**

1.1 Backgrounds and Motivations 1

1.2 Related Works 3

1.2.1 Point-based Registration 3

1.2.2 Line-based Registration 3

1.2.3 Plane-based Registration 4

1.3 Contributions 6

1.4 Organization 8

제 2 장 Preliminaries **10**

2.1 Plane Patch 10

2.1.1 Notation for Plane Patch 10

2.1.2 Problem Formulation using Plane Patch 14

2.2 Quaternion 16

2.2.1 Basis of quaternion 16

2.3 RANSAC 18

제 3 장 Proposed Method **21**

3.1 Selection of plane patch pair 22

3.2 Evaluation of Rigid Transformation 25

3.2.1 Evaluation of Rotation Matrix based on Quaternion . . . 25

3.2.2 Evaluation of Translation Vector based on Moore-Penrose Pseudo Inverse Matrix 27

3.3 Transformation of Plane Patches 27

3.4 Mean Square Distance Function 32

3.5 Selection of Rigid Transformation 37

3.6 Optimization of Translation Vector 38

제 4 장 Simulations **40**

4.1 Preconditions for the Simulation 40

4.2 Validity of Random Iteration 41

4.3 Simulation Results 46

제 5 장 Real Experiments **49**

5.1 Environments for the Experiments 49

5.2 Extraction of plane patches from point cloud 50

5.3 Results of Real Experiments 51

제 6 장 Conclusion **52**

참고문헌 **53**

Abstract **59**

감사의 글 **61**

그림 목차

그림 1.1 Simultaneous localization and mapping 2

그림 1.2 Point Cloud Matching 4

그림 1.3 Line based registration 5

그림 1.4 Plane based registration 6

그림 1.5 Process of MUMC algorithm 7

그림 1.6 PRRUS algorithm 8

그림 2.1 Plane Patch 11

그림 2.2 Mean square distance of one plane patch 12

그림 2.3 Matrix S of one plane patch 13

그림 2.4 Problem Description 16

그림 2.5 quaternion i, j, k 17

그림 2.6 Quaternion and 3D Rotation 18

그림 3.1 Process of proposed algorithm 21

그림 3.2 Selection of plane patch pair in two frames 22

그림 3.3 Matching pairs 23

그림 3.4 Iteration process 24

그림 3.5 Transformation of Plane 28
그림 3.6 Transformation of set of plane patches 31
그림 3.7 Similarity evaluated by normal vector and distance to origin 33
그림 3.8 graph of arccos function 34
그림 3.9 graph of arccos function 35
그림 3.10 Mean square distance between two plane patches 36
그림 3.11 Mean square distance between a plane patch and a set of plane patches 36
그림 3.12 Mean square distance between two sets of plane patches . . 37
그림 3.13 Calibration Translation Vector $\mathbf{t}_{opt}$ for single plane patch . 38
그림 3.14 Calibration Translation Vector $\mathbf{t}_{opt}$ in two frames 39

그림 4.1 Example of simulation 41
그림 4.2 Example of frame A 42
그림 4.3 Success rate of random iteration 44
그림 4.4 Execution time of random iteration 45
그림 4.5 Common Frame number 5-100 47
그림 4.6 Simulation Results-Execution time 48
그림 4.7 Simulation Results-Success rate 48

그림 5.1 Environment of real experiments 49

그림 5.2 3D point cloud extracted from the RGB-D image 50

그림 5.3 Results of real experiments-execution time and success rate 51

표 목차

표 2.1 Symbols 14

표 2.2 Frame A, B, A' and symbols 15

표 4.1 Comparison between random iteration and considering all the cases. k=4000 44

표 4.2 Frame number 50, Outlier rate 70% 47

표 5.1 Success rate and Execution time 51

제 1 장 Introduction

1.1 Backgrounds and Motivations

로봇산업이 빠른 속도로 발전을 하며 로봇은 더 이상 고정된 공간에서 단순한 작업을 반복하는 도구가 아니게 되었다. 이제 로봇은 실내, 실외 뿐만 아니라 인간이 접근할 수 없는 해저, 지하, 우주 등의 다양한 공간에서 인간을 대신하여 정보를 수집하고 명령을 수행하는 대체자의 역할을 하게 되었다. 이러한 작업을 효율적으로 수행하기 위해서는 SLAM(Simultaneous Localization and Mapping)과 같이 로봇이 자율적으로 주행을 하며 동시에 주변의 환경을 인식하여 지도를 만들 수 있는 기술이 필수적이다 [1],[12],[13],[27],[28]. SLAM의 정확도는 로봇이 얻은 주변 환경에 대한 정보를 토대로 얼마나 정확하게 자신의 위치를 파악할 수 있는지에 직접적으로 영향을 받는다. 즉 두 프레임에서 얻어진 데이터를 토대로 강체변환을 추정하는 정합(Registration) 기술이 핵심적이다 [17].

쿼드롭터, 드론 등의 비행로봇이 등장하며 2차원이 아닌 3차원에서의 정합기술이 더욱 중요시 여겨지고 있다는 점, 움직이는 장애물이 있는 동적 환경에서 사용될 수 있다는 점에서 최근에는 평면 패치(Plane Patch)를 기반으로 한 정합기술이 화두가 되고 있다 [2], [8]. 평면을 이용하면 이미지 정보에서 픽셀, 혹은 Point Cloud의 단위의 정합보다 수행시간이 훨씬 짧고, 움직이는 물체가 지나가도 천장, 벽, 바닥 등의 정보를 이용하여 강인하게 정합을 수행할 수 있다는 장점이 있다.

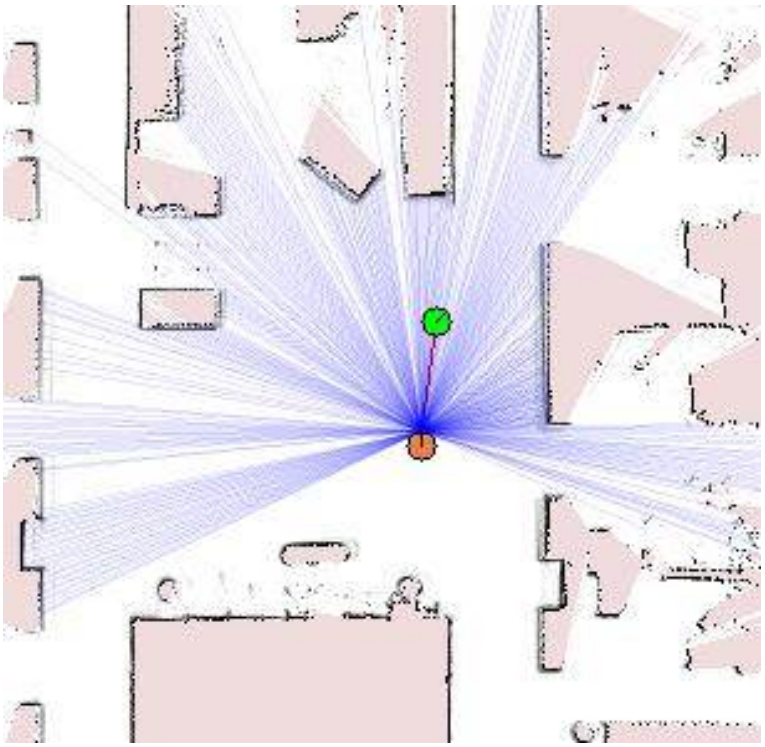


그림 1.1 Simultaneous localization and mapping

정합을 수행할 때 결과값으로 나오는 강체 변환의 정확도는 한 프레임의 점/분포가 다른 프레임의 어떤 점/분포와 대응이 되었는지 여부, 즉 올바른 대응관계(correspondence)에 직접적으로 영향을 받는다. 그렇기 때문에 두 프레임 사이의 정확한 대응관계를 구하는 문제는 정합 분야에서 항상 주요한 문제로 대두되어 왔었다. 평면 패치로 정합을 하는 경우에도 마찬가지인데, 이 대응관계를 빠르고 효율적으로 구하여 효과적으로 평면 기반 정합을 수행할 수 있는 방법이 고안하다가 이 연구를 진행하게 되었다.

1.2 Related Works

기존에 나와있는 정합기술들 중 대표적인 것들을 점 기반 정합, 선 기반 정합, 그리고 평면 기반 정합으로 나누어 설명하였다.

1.2.1 Point-based Registration

3차원 정합 알고리즘으로 가장 대표적인 알고리즘은 Iterative Closest Point(ICP) [3], [5]이다. ICP는 점과 점을 직접 대응시키는(point-to-point) 알고리즘으로, 최근점(closest point)를 대응점(corresponding point)로 대응시킨다. 즉 Euclidian distance를 기준으로 correspondence를 결정하는 것이다. 그리고 대응점 간의 거리 제곱의 총합으로 정의된 목적함수(objective function)의 값을 반복적으로 최적화하여 목적함수의 최소값을 구하고 그 값에 대응되는 변환을 추정한다. 한편, 또 하나의 대표적인 알고리즘으로 Normal Distributions Transform(NDT)가 있다 [4], [19]. NDT는 점 대 분포(point-to-distribution) 방식으로 매칭하는 알고리즘으로, 두 스캔 중 고정된 스캔을 3차원 격자(grid)로 나눈 뒤 각 칸(cell)에 포함된 점들의 평균과 공분산의 정규분포식으로 근사한 뒤 다른 스캔과 이 분포의 점수함수(score function)가 최대가 되도록 하는 방식이다.

1.2.2 Line-based Registration

Line feature를 이용한 정합에 대한 연구로는 line과 corner를 복합적으로 정합하여 visual SLAM이 있다 [6]. 로봇이 이동하며 얻은 이미지 파일에서 line de-

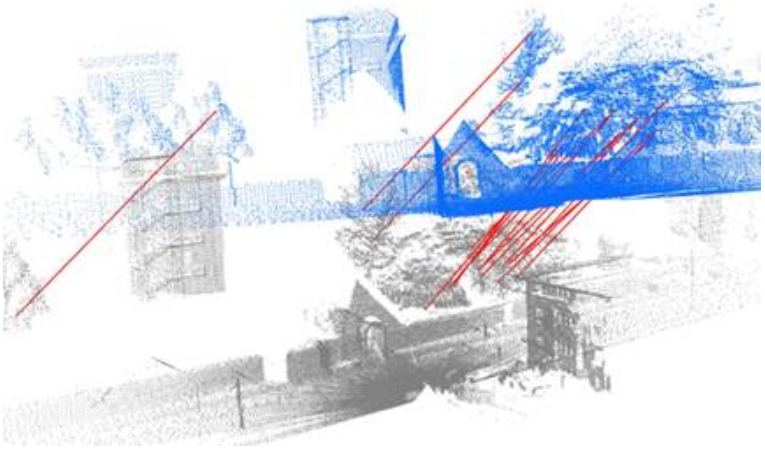


그림 1.2 Point Cloud Matching

tection과 tracking을 이용하여 line을 추출한 후 수직선과 수평선을 landmark로 잡아 이를 매칭 시키며 correspondence를 결정한다.

1.2.3 Plane-based Registration

최근에는 평면 패치(plane patch)를 기반으로 하는 정합(registration) 기법도 많이 사용되고 있다 [7], [8], [9], [10], [11]. 이 기법은 분포 대 분포(distribution-to-distribution) 방식으로 RGB-D 이미지 또는 3차원 Point Cloud으로 이루어진 두 프레임에서 평면의 법선벡터(normal vector)와 원점까지의 거리(distance to origin)를 각각 추출한 다음, 두 프레임의 평면들의 집합을 가장 잘 들어맞게 하는 변환을 추정하는 방식이다. 뿐만 아니라 평면 패치와 점을 동시에 사용하여 매칭을 시키는 기법도 연구되고 있다 [18]. 평면을 추출하는 기법에는 주로 PCA 혹은 Region Growing에 기반한 알고리즘이 있다 [20],[21],[23]. 또한 추출된 평면

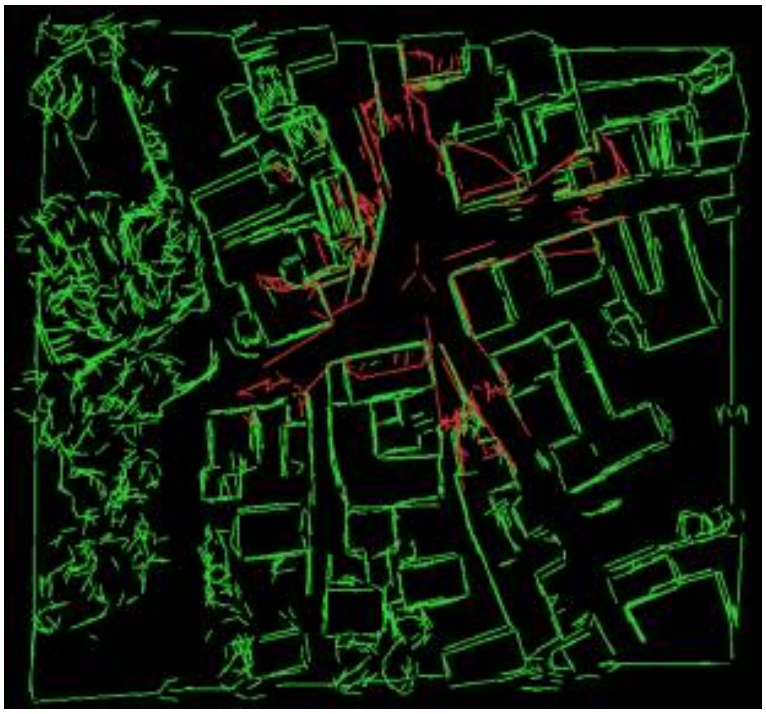


그림 1.3 Line based registration

에 대한 분석 및 보정작업을 통한 추출 기법의 개선에 연구도 진행되고 있다 [22], [26], [29]. 평면 패치를 기반으로 한 정합 기법은 많은 수의 점을 하나의 평면으로 묶어서 보기 때문에 모든 점의 정보를 사용하는 ICP나 NDT에 비하여 매우 빠르기 때문에 실시간으로 정합이 필요한 작업에 이용될 수 있다. 또한 실내 환경에서 이미지를 얻을 경우 천장이나 벽, 바닥 등이 잘 추출되기 때문에 더욱 좋은 변환결과를 얻어낼 수 있다.

그러나 실제 환경에서 정합을 하는 경우 대응관계를 알고있지 않는 경우가 훨씬 많고 이 때는 적절한 대응관계를 찾기 위한 평면의 유사도를 다양한 방법으로 검

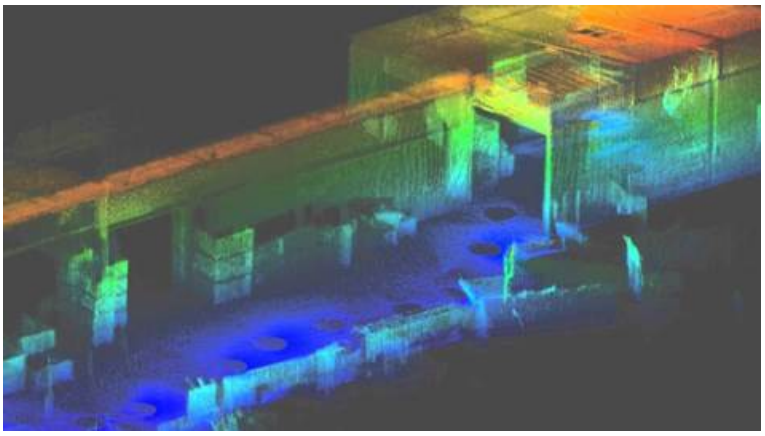


그림 1.4 Plane based registration

증하는 Minimally Uncertain Maximal Consensus(MUMC) [10] 등의 알고리즘이 있다. 이 알고리즘은 다양한 기준을 가지고 그 기준들을 모두 만족하는 평면들의 쌍(pair)을 찾기 때문에 수행시간이 길다는 단점이 있다. 이를 개선하기 위하여 단위 구의 회전을 기반으로 하는 Plane Registration based on Rotation of a Unit Sphere(PRRUS) [11] 의 알고리즘도 있다. 그러나 이 알고리즘 역시 대응관계를 찾기 위해 1-자유도(one degree of freedom)가 더 필요하다는 단점이 있다.

1.3 Contributions

우리는 이 단점을 해결하기 위하여 Random Sample Consensus(RANSAC) 알고리즘 [14], [15], [16]을 응용하여 평면의 correspondence를 구하였다. 이 논문에서 제안한 기법은 임의로 평면 패치들의 대응관계를 결정한 후 그 관계에 부합하는 강체변환을 구하여 두 프레임을 합친 결과를 비교한다. 이 과정을 반복적으로 수

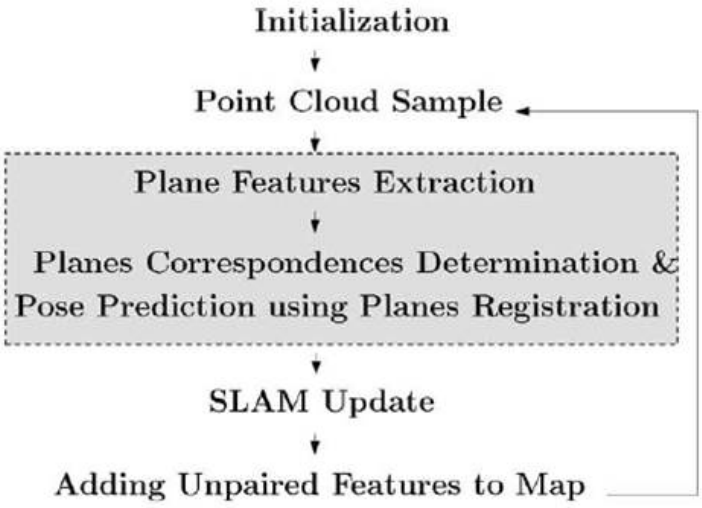


그림 1.5 Process of MUMC algorithm

행하여 가장 이상점(outlier)의 개수가 작은 변환을 최종 결과값으로 이끌어낸다. 제시한 새로운 방법으로 시뮬레이션 및 실제 환경에서의 실험을 수행한 결과, 빠른 수행속도와 정확성을 확인할 수 있었다.

이러한 단점을 보완하기 위해 우리는 두 평면의 거리제곱평균 함수를 이용하여 유사도를 결정하였고 이를 기준으로 다음과 같이 알고리즘을 수행한다. 먼저 두 프레임에서 임의로 평면의 대응쌍을 지정한 다음 그 쌍들에 대응되는 강체변환을 쿼터니온 및 pseudo inverse Matrix를 이용하여 반복적으로 구한다. 구해진 각각의 강체변환을 적용하였을 때의 두 프레임의 유사도를 비교하여 가장 유사하게 매칭시키는 강체변환을 1차 결과로 반환한다. 그 다음 두 프레임의 유사도를 최적화 시키는 보정 평행이동벡터를 1차 강체변환에 더하여 최종 강체변환을 구한다. 이 논문에서 제시한 알고리즘은 이미지의 모든 정보를 사용하지 않고 평면객체

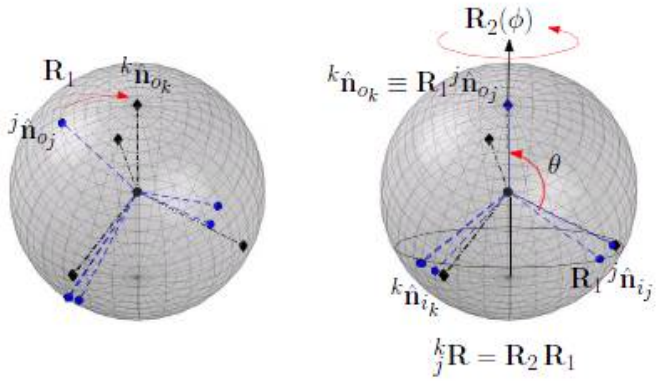


그림 1.6 PRRUS algorithm

만을 이용한다는 점과 RANSAC 알고리즘을 기반으로 동작한다는 점에서 빠른 수행속도와 outlier에 대한 강인성을 가진다

1.4 Organization

본 논문은 다음과 같이 구성되어 있다. 2장에서는 본 논문에서 제안한 기법을 소개하기에 앞서 알아야 할 개념들에 대하여 기술한다. 평면 기반 정합과정을 명확하게 설명하기 위하여 우선 평면 패치를 수학적으로 기술한다. 강체변환을 구하는데 사용되는 사원수(Quaternion)의 개념에 대하여 설명하고, 평면의 임의 대응쌍을 선택하는 기법의 토대가 되는 RANdom SAmple Consensus(RANSAC)의 개념도 다룬다. 3장에서는 본격적으로 본 논문에서 제안한 알고리즘에 대하여 설명한다. 정합을 수행하고자 하는 두 프레임에서 각각 임의로 평면의 쌍을 선택하고, 이를 기반으로 강체변환을 계산한 다음 한 프레임을 변환하여 다른 프레임과 매

칭을 시키는 과정에 대하여 설명한다. 그리고 평면의 유사도를 판단하는 기준으로 사용하는 거리제곱평균 함수의 개념을 수학적으로 기술한다. 매칭을 반복적으로 수행한 후 가장 적합한 강체변환을 구하고, 보정 평행이동 벡터를 이용하여 최적화시키는 것으로 정합을 마무리하는 과정까지 설명한다. 4장과 5장에서는 제안한 알고리즘을 시뮬레이션 및 실제 환경에서의 실험을 통해 검증하는 내용을 다루고 6장에서는 본 논문의 결론과 의의를 다룬다.

제 2 장 Preliminaries

이 장에서는 본 논문의 알고리즘을 이해하기 위하여 알아야 하는 개념들을 소개한다. 평면 패치를 수학적으로 기술하고 본 논문에서 해결하려는 문제를 수식적으로 표현한다. 그 다음 본 논문에서 사용되는 사원수와 RANSAC에 대하여 설명한다.

2.1 Plane Patch

2.1.1 Notation for Plane Patch

이 절에서는 정합의 기본 단위가 되는 하나의 평면 패치를 수학적으로 정의한다. 여기서 하나의 평면 패치는 3차원 공간에서의 여러 점들의 집합으로 이 점들을 가장 잘 나타내는 평면을 의미한다. 이를 수학적으로 기술하면 다음과 같다. 먼저 3차원 공간의 점들의 좌표는 모두 3행 1열의 벡터로 나타내기로 하자. 원점이 O인 3차원 좌표계 상의 점들의 집합 $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_k\}$으로 이루어진 평면패치 p는 집합 $p = \{\mathbf{x}|\mathbf{n}^T\mathbf{x} = d\}$의 형태로 나타낼 수 있다. 여기서 $\mathbf{n}$은 p의 법선 벡터, d는 p와 점 O 사이의 거리를 나타낸다. 실제로 평면을 추출할 경우에는 p의 모든 원소들이 조건 $\mathbf{n}^T\mathbf{x} = d$를 만족하지는 않기 때문에 우리는 평면에 속하는 점들이 그 평면과 얼마나 떨어져 있는지를 구하여 이후 절에 나오는 두 프레임의 유사도 측정에 사용한다. 유사도를 수학적으로 구하기 위하여 평면 p의 점들의 평균, 공분산행렬

및 거리제곱평균을 정의한다.

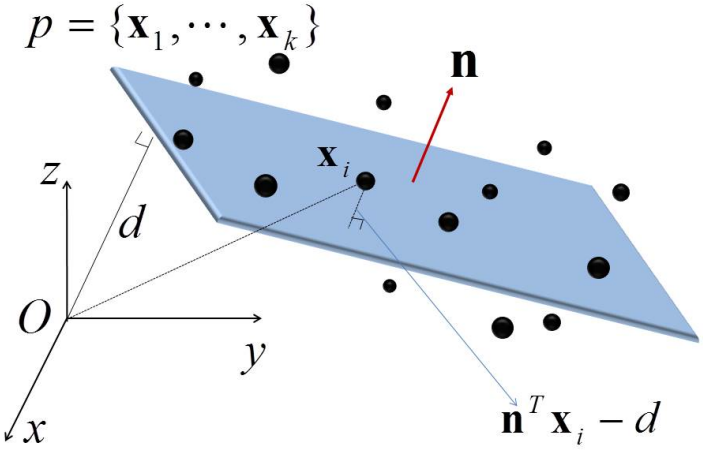


그림 2.1 Plane Patch

p의 원소의 개수를 k라 할 때, p의 원소인 점들의 평균 $\boldsymbol{\mu}$는 각 점들의 좌표의 평균을 구하여 얻을 수 있다.

$$\boldsymbol{\mu} = \sum_{i=1}^{k} \mathbf{x}_i \tag{2.1}$$

평면패치 p의 평균을 구하면 이제 p에 속하는 점들이 평균과 얼마나 떨어져 있는지를 공분산행렬 $\mathbf{S}$를 사용하며 계산할 수 있다. $\mathbf{S}$는 $\mathbf{x}_i$와 $\boldsymbol{\mu}$의 성분별 차이를 더하여 평균을 낸 식으로 아래와 같이 표현할 수 있다.

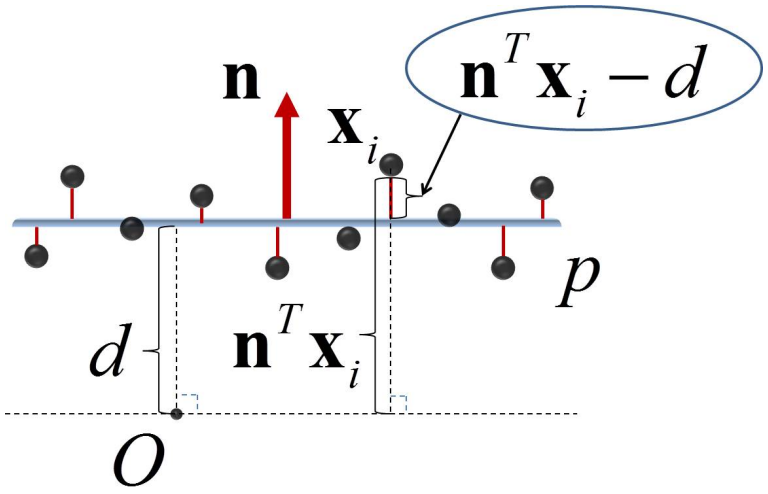


그림 2.2 Mean square distance of one plane patch

$$\mathbf{S} = \sum_{i=1}^{k} \frac{(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T}{k} \tag{2.2}$$

식 2.2에 식 2.1를 대입하여 정리하면 다음과 같이 정리할 수 있다.

$$\mathbf{S} = \sum_{i=1}^{k} \frac{\mathbf{x}_i\mathbf{x}_i{}^T - \boldsymbol{\mu}\boldsymbol{\mu}^T}{k} \tag{2.3}$$

한편, p에 속하는 점들이 평면과 얼마나 떨어져 있는지는 거리제곱평균 msd을 이용하여 나타낼 수 있다. p의 원소들에 대한 msd는 다음과 같이 주어진다.

$$msd = \sum_{i=1}^{k} \frac{(\mathbf{x}_i{}^T\mathbf{n} - d)^2}{k} \tag{2.4}$$

공분산 행렬 **S**와 거리제곱평균 msd는 법선벡터 **n**을 사용하여 서로 간의 관계

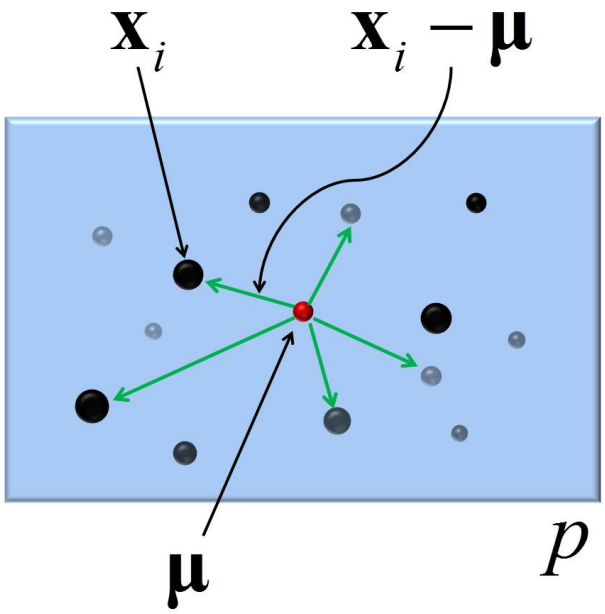


그림 2.3 Matrix S of one plane patch

를 수식으로 나타낼 수 있다. 식 2.4에서 $(\mathbf{x}_i{}^T\mathbf{n} - d)$가 scalar 값인 것을 이용하여 $(\mathbf{x}_i{}^T\mathbf{n} - d)^2$을 $(\mathbf{x}_i{}^T\mathbf{n} - d)^T(\mathbf{x}_i{}^T\mathbf{n} - d)$와 같이 바꾸어 표현할 수 있고 이를 수식으로 전개하면 다음과 같은 관계식이 나온다.

$$msd = \mathbf{n}^T\mathbf{S}\mathbf{n} \tag{2.5}$$

이 값들을 한 번에 정의하기 위하여 하나의 평면을 $p(\mathbf{n}, d, \boldsymbol{\mu}, \mathbf{S}, msd, k)$와 같이 표기한다. 간단히 나타낼 경우에는 괄호 안의 내용을 생략하고 p와 같이 표기한다.

변수	의미
p	하나의 평면패치
P	평면패치들의 집합
O	좌표계의 원점
$\mathbf{x}$	평면에 속하는 하나의 점
$\mathbf{n}$	평면의 법선벡터
d	평면과 원점과의 거리
k	평면에 속하는 점들의 개수
$\boldsymbol{\mu}$	평면에 속하는 점들의 평균
$\mathbf{S}$	평면의 공분산 행렬
msd	평면에 속하는 점들의 거리제곱평균

표 2.1 Symbols

2.1.2 Problem Formulation using Plane Patch

이제 본 논문에서 해결하려고 하는 문제를 수학적으로 기술한다. 본 논문에서는 평면 패치를 기반으로 정합을 하기 때문에 입력으로 들어온 이미지 파일에서 평면 패치들을 추출하게 된다. 즉 하나의 프레임을 여러 개의 평면 패치들의 집합으로 표현한다. 프레임 A를 나타내는 평면 패치들의 집합을 ${}^{A}P$라 할 때 ${}^{A}P = \{{}^{A}p_1, {}^{A}p_2, ..., {}^{A}p_{N_A}\}$와 같이 표현할 수 있다. 여기서 N_A는 프레임 A에 있는

평면 패치들의 개수이고 대문자 P는 평면 패치들의 집합, 소문자 p는 하나의 평면 패치를 나타낸다. 좌상단의 첨자 A는 속해있는 프레임을 나타내며 ${}^{A}P$의 원소들은 ${}^{A}p_i = \{\mathbf{x}|{}^{A}\mathbf{n}_i^T\mathbf{x} = {}^{A}d_i\}$과 같이 표현된다. 이 때 $i = 1, 2, ..., N_A$이다. 같은 방식으로 프레임 B를 나타내는 평면 패치들의 집합은 ${}^{B}P = \{{}^{B}p_1, {}^{B}p_2, ..., {}^{B}p_{N_B}\}$와 같이 나타낼 수 있고, 각각의 평면들은 ${}^{B}p_i = \{\mathbf{x}|{}^{B}\mathbf{n}_i^T\mathbf{x} = {}^{B}d_i\}$와 같이 나타낸다. 이를 2.1.1의 표기법을 적용하여 ${}^{A}p_i({}^{A}\mathbf{n}_i, {}^{A}d_i, {}^{A}\boldsymbol{\mu}_i, {}^{A}\mathbf{S}_i, {}^{A}msd_i, {}^{A}k_i)$와 같이 표현한다. 그 외에도 앞으로 많이 쓰이게 될 표현들을 표 2.2에 정리해놓았다.

	프레임 A	프레임 B	프레임 A'
평면 패치들의 집합	${}^{A}P$	${}^{B}P$	${}^{A'}P$
평면 패치의 개수	N_A	N_B	$N_{A'}$
평면 패치	${}^{A}p_i$	${}^{B}p_i$	${}^{A'}p_i$
좌표계의 원점	${}^{A}O$	${}^{B}O$	${}^{A'}O$

표 2.2 Frame A, B, A' and symbols

우리가 해야 할 일은 프레임 A에서 본 프레임 B의 회전변환 행렬 ${}^{A}_{B}\mathbf{R}$및 평행이동 벡터 ${}^{A}_{B}\mathbf{t}$의 추정치를 구하는 것이다. 즉 프레임 B에 ${}^{A}_{B}\mathbf{R}$와 ${}^{A}_{B}\mathbf{t}$를 적용하여 얻은 새로운 프레임 A'이 원래의 프레임 A와 가장 유사하게 만드는 ${}^{A}_{B}\mathbf{R}$, ${}^{A}_{B}\mathbf{t}$를 구해야 하는 것이다. 본 논문에서는 프레임 A'과 A의 유사도를 측정하기 위하여 '두 프레임 사이의 거리제곱함수'를 정의하였고 이는 3.4절에 기술된다.

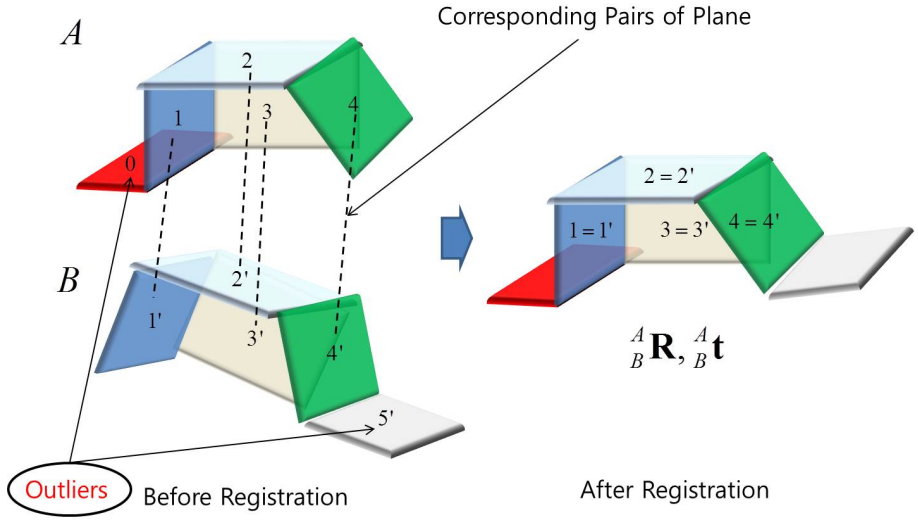


그림 2.4 Problem Description

2.2 Quaternion

2.2.1 Basis of quaternion

복소수의 확장으로서 윌리엄 해밀턴(William Hamilton)이 생각해 낸 수로써, 일종의 벡터이다. 복소수 z가 $i^2 = -1$이 되는 수 i와 실수 a, b에 대하여 $z = a + bi$의 형태로 표현 되듯이 사원수 α는 $i^2 = j^2 = k^2 = -1$이 되는 i, j, k와 실수 a, b, c, d에 대하여 $\alpha = a + bi + cj + dk$의 형태로 나타내어진다. 사원수에서 사칙연산은 가능하지만 곱셈의 교환법칙은 성립하지 않는다는 특성이 있다. 또한 사원수의 허수부들은 다음의 연관성을 가진다 [30].

사원수를 표현하는 방법에는 $\mathbf{q} = [s,\ \mathbf{v}]$, $\mathbf{q} = [s + \mathbf{v}]$와 같이 두 가지가 존재한다. 앞의 표현은 실수와 벡터를 따로 표현한 것이고, 뒤의 표현은 실수와 벡터의

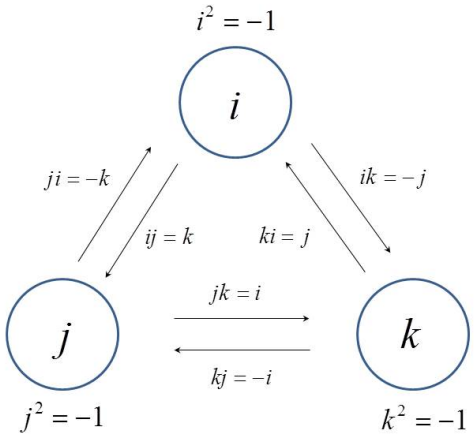


그림 2.5 quaternion i, j, k

합으로 표현한 것이다. 여기서 벡터부분은 복소수의 세 허수부를 카르테시안 벡터의 형태로 나타낸 것이다.

$$\mathbf{q} = [s + \mathbf{v}] = [s + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}] \tag{2.6}$$

이를 이용하여 두 사원수 $\mathbf{q}_1 = [s_1 + \mathbf{v}_1]$, $\mathbf{q}_2 = [s_2 + \mathbf{v}_2]$의 곱을 정의할 수 있고 다음의 관계식을 가진다.

$$\mathbf{q}_1\mathbf{q}_2 = s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2 + s_1\mathbf{v}_2 + s_2\mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2 \tag{2.7}$$

복소수에서 공액 복소수(conjugate complex)가 있듯이 사원수에도 공액 사원

수(conjugate quaternion)이 존재한다. 사원수 $\mathbf{q}$의 공액 사원수 $\mathbf{q}^*$는 $\mathbf{q}^* = [s, -\mathbf{v}]$와 같이 나타낼 수 있다. 사원수 $\mathbf{q}$와 그의 공액 사원수 $\mathbf{q}^*$의 곱은 $\mathbf{q}$의 크기의 제곱과 같다. 즉 $\mathbf{q}\mathbf{q}^* = q^2$이다.

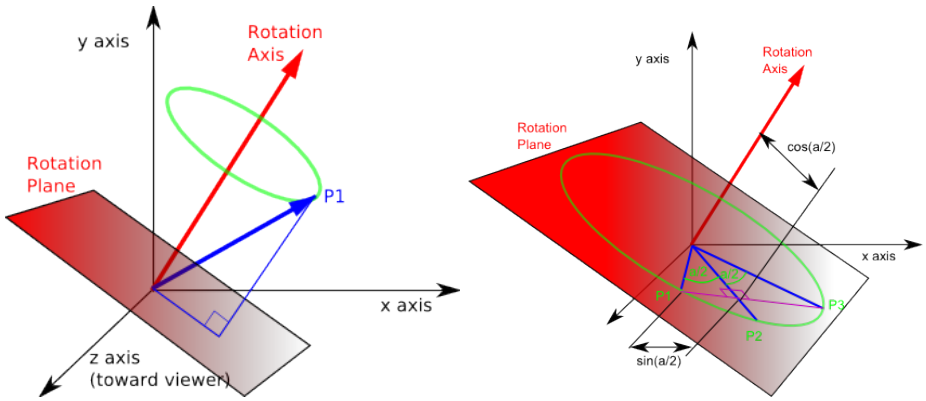


그림 2.6 Quaternion and 3D Rotation

2.3 RANSAC

대부분의 model fitting 과정에서 가장 큰 문제가 되는 것은 outlier의 처리 문제이다. outlier란 model과 크게 동떨어진 곳에 위치하는 데이터들을 말하는데 이는 model fitting을 할 때 매우 큰 noise로 작용하여 올바른 결과를 산출하는데 방해가 된다. 이를 방지하려면 outlier를 무시하거나 적은 가중치를 적용하여 알고리즘에 끼치는 악영향을 감소시켜야 한다.

이러한 outlier를 효과적으로 처리할 수 있는 model fitting 알고리즘 중 하나가 바로 RANSAC이다. RANSAC은 RANdom SAmple Consensus의 약자로 가장 많은 수의 데이터들로부터 지지를 받는 모델을 선택하는 방법이다. 기본적인 idea

는 다음과 같다. 우선 전체 데이터 중에 무작위로 몇 개의 샘플 데이터를 뽑는다. 그리고 그 샘플 데이터들을 만족하는 model parameter를 구한 다음 구한 모델과 가까이에 있는 데이터의 개수를 센다. 그 개수가 크면 모델을 기억해두다가 같은 과정을 N번 반복 후 가장 지지하는 데이터수가 많은 모델을 최종 결과로 반환한다. 이를 의사코드(pseudo-code)로 표현하면 다음과 같다 [14], [15], [16].

Algorithm 1 RANSAC method

1: Select randomly the minimum number of points required to determine the model parameters.
2: Solve for the parameters of the model.
3: Determine how many points from the set of all points fit with a predefined tolerance ϵ.
4: If the fraction of the number of inliers over the total number points in the set exceeds a predefined threshold τ, re-estimate the model parameters using all the identified inliers and terminate.
5: Otherwise, repeat steps 1 through 4 (maximum of N times).

인자 t와 d의 값은 해당 응용(application)과 data set의 요구치로부터 결정되며 실험적 추론에 근거한다. 그러나 인자 k(반복 회수)는 이론적 추론으로 결정할 수 있다. 특정 반복 회수에서 RANSAC 알고리즘이 모델 인자가 추정된 m개의 점을 선택했을 때 inlier만을 선택할 수 있는 확률을 η_0라고 하자. 즉 는 해당 알고리즘이 쓸모있는 결과를 만들어낼 수 있는 확률을 의미한다. 하나의 point를 선택했을 때 inlier일 확률을 ϵ이라 하자.

$$\epsilon = \frac{Number\ of\ inliers\ in\ data}{Number\ of\ points} \tag{2.8}$$

ϵ은 일반적으로 사전에 알려지지는 않고 추정치를 통하여 얻을 수 있다. 모델을

유추하는 데 필요한 m개의 점이 각각 독립적으로 선택되었다고 가정하면 m개의 점이 inlier일 확률은 ϵ^m이고, 최소한 1개의 점이 inlier가 아닐 확률은 $1-\epsilon^m$과 같다. 따라서 k번의 iteration 동안 모두 inlier만 있는 모델이 선택되지 않을 확률은 $(1-\epsilon^m)^k$이고 이는 $1-\eta_0$와 같다. 즉 다음과 같은 관계식이 성립하게 된다.

$$1-\eta_0 = (1-\epsilon^m)^k \tag{2.9}$$

그렇기 때문에 confidence η_0이상을 만족하기위하여 수행해야 하는 iteration의 회수 k는 다음 식과 같이 주어진다.

$$k \geq \frac{\log(1-\eta_0)}{\log(1-\epsilon^m)} \tag{2.10}$$

제 3 장 Proposed Method

3장에서는 본격적으로 두 프레임의 평면패치들의 집합을 정합하는 과정을 다룬다. 정합하고자 하는 두 프레임에서 각각 임의의 평면 대응쌍을 설정한 후 이를 기반으로 강체변환을 일차적으로 구한다. 이 과정을 반복적으로 수행하며 구한 각각의 변환으로 두 프레임을 정합하였을 때 매칭되는 정도를 거리제곱함수로 측정하여 이 값을 최소화 시키는 강체변환을 최종 후보로 선택한다. 그리고 정해진 변환에 대하여 평행이동벡터를 최적화시켜 정합을 마무리 한다.

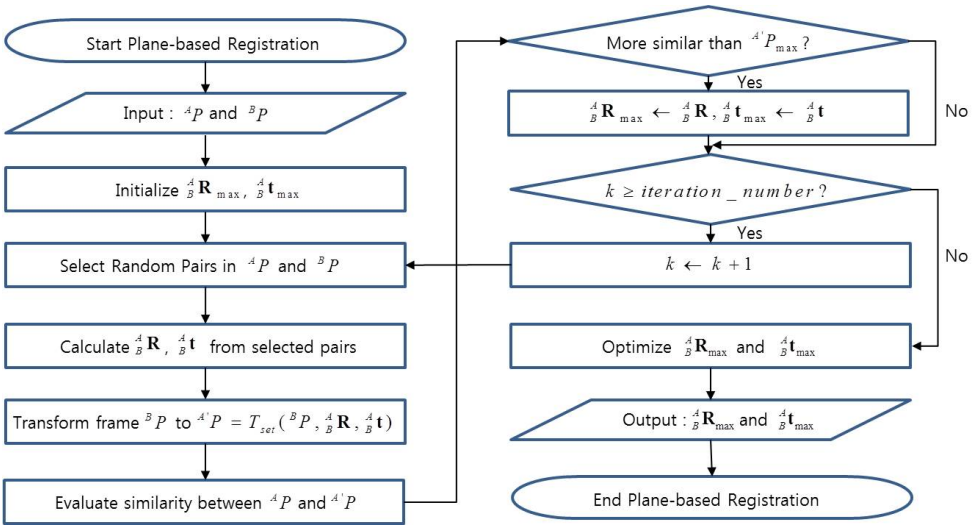


그림 3.1 Process of proposed algorithm

3.1 Selection of plane patch pair

정합을 하고자 하는 두 프레임 사이의 회전변환 행렬 및 평행이동 벡터를 구하려면 각 프레임에서 서로 대응되는 특징점들을 지정해주어야 한다. 본 논문에서는 평면 패치를 기반으로 정합을 수행하므로 정합을 하기 위하여 기본적으로 각 프레임에서 서로 대응되는 평면 패치들을 지정해주어야 한다. 제안한 기법은 이 대응되는 쌍들을 임의로 그리고 반복적으로 지정하며 가장 적절한 대응쌍 그리고 그에 해당하는 회전변환 행렬 및 평행이동 벡터를 찾는다. 이를 수학적으로 기술하면 다음과 같다.

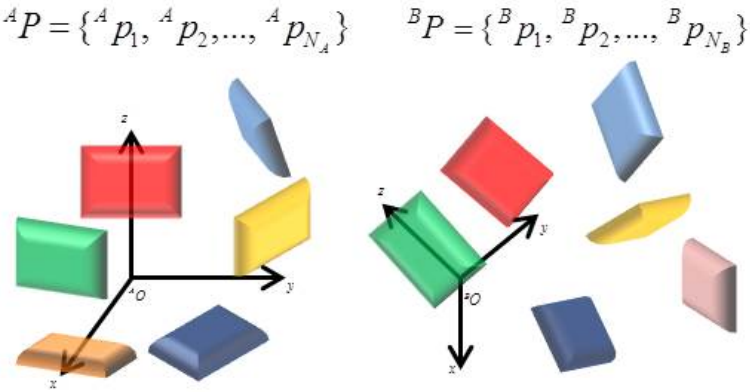
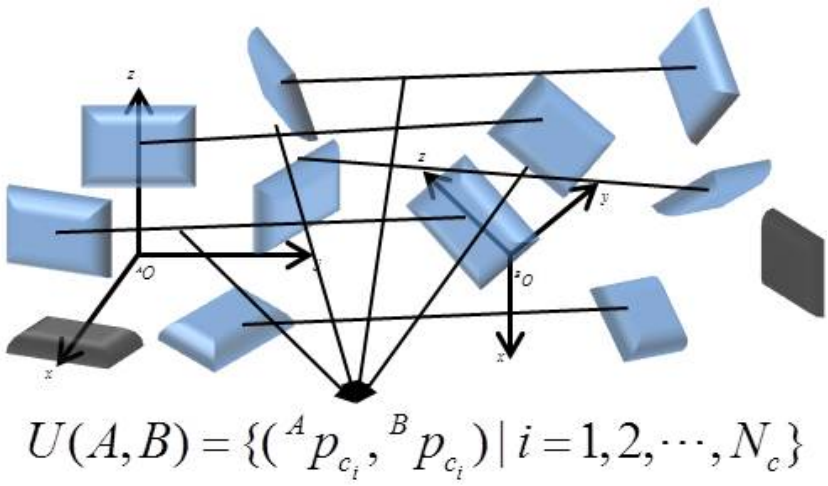


그림 3.2 Selection of plane patch pair in two frames

그림 3.2과 같이 프레임 A를 나타내는 평면패치들의 집합을 ${}^{A}P = \{{}^{A}p_1, {}^{A}p_2, ..., {}^{A}p_{N_A}\}$, 그리고 프레임 B를 나타내는 평면패치들의 집합 ${}^{B}P = \{{}^{B}p_1, {}^{B}p_2, ..., {}^{B}p_{N_B}\}$라 하자. 이 때 두 프레임 간 실제로 대응되고 있는 쌍들의 집합을 그림 3.3과 같이 $U(A,B) = \{({}^{A}p_{c_i}, {}^{B}p_{c_i}) | i = 1, 2, ..., N_c\}$와 같이 나타낼 수 있다. 즉 ${}^{A}p_{c_1}$

와 ${}^{B}p_{c_1}$, ${}^{A}p_{c_2}$와 ${}^{B}p_{c_2}$, ..., ${}^{A}p_{c_{N_c}}$와 ${}^{B}p_{c_{N_c}}$가 서로 대응되는 것이다. 여기서 c_i는 대응되는 평면 패치들의 인덱스를 나타내며 1부터 $\min(N_A, N_B)$사이의 값을 가진다. 또한 $i \neq j$이면 $c_i \neq c_j$으로 하나의 평면패치가 중복하여 대응쌍을 이루지 않는다고 가정한다. 이제 반복적으로 평면 패치의 임의 대응쌍을 지정하는 과정은 $U_{rand}(A,B) = \{({}^{A}p_{r_i}, {}^{B}p_{r_i}) | i = 1, 2, ..., N_{rand}\}$를 만드는 과정으로 나타낼 수 있다. 여기서 첨자 r_i는 임의로 지정되는 1과 N_c 사이의 값이며 임의 대응쌍 역시 중복되어 선택되지 않기 때문에 $i \neq j$이면 $r_i \neq r_j$을 만족한다.



$$U(A,B) = \{({}^{A}p_{c_i}, {}^{B}p_{c_i}) \mid i = 1, 2, \cdots, N_c\}$$

그림 3.3 Matching pairs

이제 우리는 다음과 같은 과정을 통하여 1차 회전변환 행렬 ${}^{A}_{B}\mathbf{R}_1$ 및 1차 평행이동 벡터 ${}^{A}_{B}\mathbf{t}_1$을 추정한다.

1) ${}^{A}_{B}\mathbf{R}_1 \leftarrow \mathbf{I}_{3\times3}$, ${}^{A}_{B}\mathbf{t}_1 \leftarrow \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$, $D_{min} \leftarrow \infty$로 초기화한다. 여기서 $\mathbf{I}_{3\times3}$

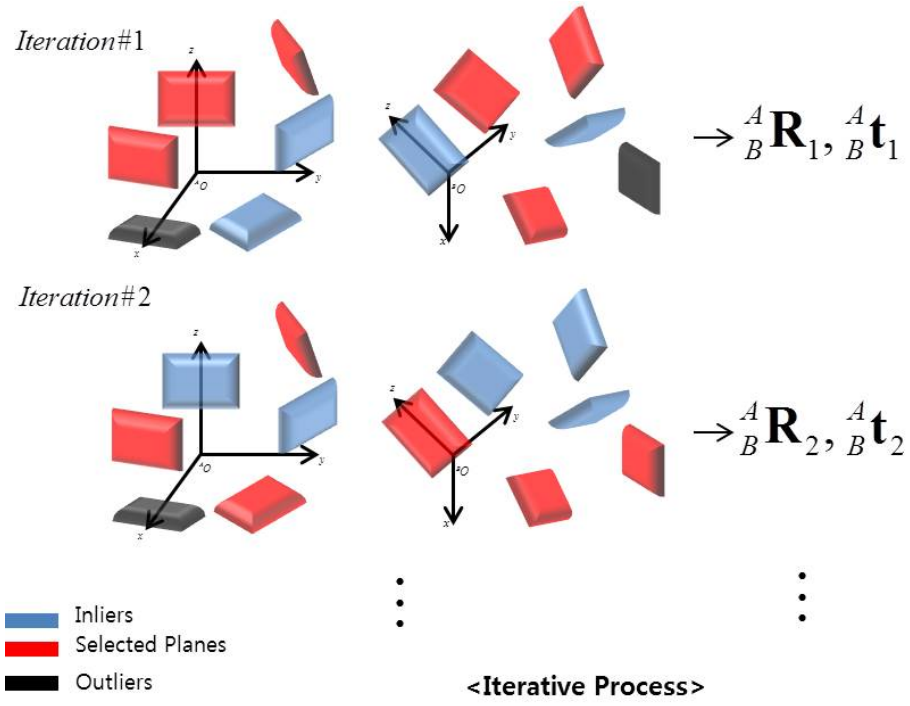


그림 3.4 Iteration process

는 3행 3열의 단위행렬이다.

2) ${}^{A}P$, ${}^{B}P$에서 각각 임의로 N_{rand}개의 평면을 선택하여 평면패치들의 임의 대응쌍 $U_{rand}(A,B) = \{({}^{A}p_{r_i}, {}^{B}p_{r_i}) | i = 1, 2, ..., N_{rand}\}$을 만든다.

3) $U_{rand}(A,B)$에 대응되는 ${}^{A}_{B}\mathbf{R}_{rand}$ 및 ${}^{A}_{B}\mathbf{t}_{rand}$를 구한다.

4) ${}^{B}P$를 ${}^{A'}P = T_{set}({}^{B}P, {}^{A}_{B}\mathbf{R}_{rand}, {}^{A}_{B}\mathbf{t}_{rand})$로 변환한 후 $D_{plane}({}^{A'}P|{}^{A}P)$를 구한다. $D_{plane}({}^{A'}P|{}^{A}P)$가 D_{min}보다 작을 경우 $D_{min} \leftarrow D_{plane}({}^{A'}P|{}^{A}P)$, ${}^{A}_{B}\mathbf{R}_1 \leftarrow {}^{A}_{B}\mathbf{R}_{rand}$, ${}^{A}_{B}\mathbf{t}_1 \leftarrow {}^{A}_{B}\mathbf{t}_{rand}$로 D_{min}, ${}^{A}_{B}\mathbf{R}_1$, ${}^{A}_{B}\mathbf{t}_1$의 값을 갱신한다.

5) 2)부터 4)까지의 과정을 L번 반복한 후 ${}^{A}_{B}\mathbf{R}_1$, ${}^{A}_{B}\mathbf{t}_1$를 결과값으로 반환한다.

3.2 Evaluation of Rigid Transformation

반복적으로 평면 패치들의 임의 대응쌍을 정하는 과정에서 대응쌍이 일단 결정되면 그에 상응하는 회전변환 행렬과 평행이동 벡터를 수학적으로 구할 수 있다. 이 절에서는 쿼터니온을 기반으로 회전변환 행렬을 구하는 과정과 Moore-Penrose Pseudo Inverse Matrix를 이용하여 평행이동 벡터를 구하는 과정을 다룰 것이다.

3.2.1 Evaluation of Rotation Matrix based on Quaternion

정합하고자 하는 두 프레임에서 얻은 평면들의 대응쌍을 알고 있을 경우 쿼터니온의 개념을 사용하여 두 프레임 간의 회전변환 행렬을 구할 수 있다. 3차원 공간에서의 점 $\mathbf{x}$은 $\breve{\mathbf{x}} = \begin{bmatrix} 0 & \mathbf{x}^T \end{bmatrix}^T$ 와 같이 쿼터니온의 형태로 표현할 수 있다. 회전변환 행렬 $\mathbf{R}$에 대응하는 단위 쿼터니온을 $\breve{\mathbf{q}}$라 할 때 $\mathbf{x}$에 $\mathbf{R}$을 곱하는 연산은 $\mathbf{x}^T$의 양 옆에 $\breve{\mathbf{q}}$와 $\breve{\mathbf{q}}^{-1}$를 곱하는 것과 대응된다. 여기서 $\breve{\mathbf{q}}^{-1}$는 $\breve{\mathbf{q}}$의 역쿼터니온이다.

프레임 A를 나타내는 평면패치들의 집합 ${}^{A}P = \{{}^{A}p_1, {}^{A}p_2, ..., {}^{A}p_{N_A}\}$와 프레임 B를 나타내는 평면패치들의 집합 ${}^{B}P = \{{}^{B}p_1, {}^{B}p_2, ..., {}^{B}p_{N_B}\}$ 사이의 평면 대응쌍이 주어졌을 때 두 프레임 간 회전변환 행렬 ${}^{A}_{B}\mathbf{R}$에 대응하는 쿼터니온 ${}^{A}_{B}\breve{\mathbf{q}}$을 구한다. ${}^{A}P$와 ${}^{B}P$에서 N_C개의 평면패치의 쌍이 서로 대응된다고 하자. 그러면 이 대응되는 쌍들의 집합을 $U(A,B) = \{({}^{A}p_{c_i}, {}^{B}p_{c_i}) | i = 1, 2, ..., N_c\}$와 같이 나타낼 수 있다. 평면패치의 대응쌍이 세 개 이상이어야 평행이동 벡터의 해가 유일하게 결정되기 때문에 $N_c \geq 3$이라고 가정한다. 이 때 $U(A,B)$에 대응하는 ${}^{A}_{B}\breve{\mathbf{q}}$는 아래 행렬 $\mathbf{K}$의 최대 고유값에 대응되는 고유벡터와 같다 [4].

$$\mathbf{K} = \sum_{i=1}^{N_c} \overline{\mathbf{\Phi}}^T({}^B\breve{\mathbf{n}}_i)\mathbf{\Phi}({}^A\breve{\mathbf{n}}_i) \tag{3.1}$$

식 3.1에서 ${}^A\breve{\mathbf{n}}_i$, ${}^B\breve{\mathbf{n}}_i$는 각각 ${}^Ap_i$의 법선벡터 ${}^A\mathbf{n}_i$ 및 ${}^Bp_i$의 법선벡터 ${}^B\mathbf{n}_i$에 대응하는 쿼터니온이다. 또한 행렬 $\mathbf{\Phi}(\breve{\mathbf{n}})$ 및 $\overline{\mathbf{\Phi}}(\breve{\mathbf{n}})$은 다음과 같다.

$$\mathbf{\Phi}(\breve{\mathbf{n}}) = \begin{bmatrix} n_0 & -\mathbf{n}^T \\ \mathbf{n} & n_0\mathbf{I}_3 + \mathbf{n}\times\mathbf{I}_3 \end{bmatrix} \tag{3.2}$$

$$\overline{\mathbf{\Phi}}(\breve{\mathbf{n}}) = \begin{bmatrix} n_0 & -\mathbf{n}^T \\ \mathbf{n} & n_0\mathbf{I}_3 - \mathbf{n}\times\mathbf{I}_3 \end{bmatrix} \tag{3.3}$$

식 3.2, 3.3에서 $\mathbf{n} = \begin{bmatrix} n_1 & n_2 & n_3 \end{bmatrix}^T$, $\breve{\mathbf{n}} = \begin{bmatrix} n_0 & \mathbf{n}^T \end{bmatrix}^T$이고, 벡터와 행렬의 외적 $\mathbf{n}\times\mathbf{I}_3$은 다음 식과 같다.

$$\mathbf{n}\times\mathbf{I}_3 = \begin{bmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{bmatrix} \tag{3.4}$$

식 3.1에서 구한 쿼터니온 ${}^A_B\breve{\mathbf{q}}$은 함수 $Rot : \breve{\mathbf{n}} \mapsto \mathbf{R}$을 이용해 아래와 같이 회전변환 행렬로 대응시킬 수 있다.

$$Rot(\breve{\mathbf{n}}) = ({n_0}^2 - \|\mathbf{n}\|^2)\mathbf{I}_3 + 2\mathbf{n}\mathbf{n}^T + 2n_0(\mathbf{n} \times \mathbf{I}_3) \tag{3.5}$$

식 3.5를 이용하여 ${}^A_B\mathbf{R} = Rot({}^A_B\breve{\mathbf{q}})$으로 1차 회전변환 행렬을 구한다.

3.2.2 Evaluation of Translation Vector based on Moore-Penrose Pseudo Inverse Matrix

평행이동 벡터는 아래와 같이 구할 수 있다.

$${}^A_B\mathbf{t} = (\mathbf{N}^T\mathbf{N})^{-1}\mathbf{N}^T\mathbf{d} \tag{3.6}$$

행렬 $\mathbf{N} \equiv \begin{bmatrix} {}^A\mathbf{n}_1 & \cdots & {}^A\mathbf{n}_{N_c} \end{bmatrix}^T$ 이고 벡터 $\mathbf{d} \equiv \begin{bmatrix} {}^Ad_1 - {}^B d_1 & \cdots & {}^Ad_{N_c} - {}^B d_{N_c} \end{bmatrix}^T$ 이다.

3.3 Transformation of Plane Patches

위의 과정들을 통하여 회전변환 행렬과 평행이동 벡터를 구한 다음에는 이를 적용하여 한 프레임을 다른 프레임에 매칭시킨 다음 유사도를 측정하여야 한다. 이 절에서는 회전변환 행렬과 평행이동 벡터가 결정된 상태에서 정합하고자 하는 두 개의 프레임을 하나의 프레임으로 합치는 과정에 대하여 다룬다.

우리가 이전 절에서 반복적으로 평면 패치들의 임의 대응쌍을 선택하였던 이유는 최종적으로 프레임 A와 프레임 B 사이의 강체변환을 구하기 위한 것이었다.

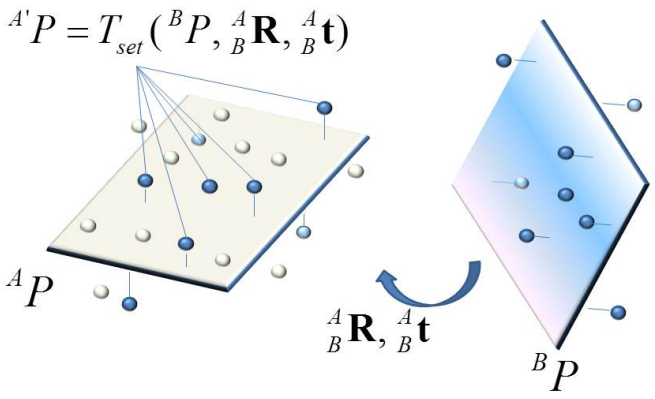


그림 3.5 Transformation of Plane

정확하게 말하자면 프레임 A에서 보았을 때 프레임 B까지의 회전변환 행렬 ${}^{A}_{B}\mathbf{R}$과 평행이동 벡터 ${}^{A}_{B}\mathbf{t}$를 구한 것이다. 그 다음에는 프레임 B에 이 변환을 적용하여 프레임 A'을 구한다. 프레임 A와 유사하다는 것을 강조하기 위하여 기호 A'을 사용하였다.

이제 프레임 A'가 프레임 B 및 ${}^{A}_{B}\mathbf{R}$, ${}^{A}_{B}\mathbf{t}$과 어떠한 관계가 있는지 수학적으로 알아본다. 우선적으로 3차원 공간을 나타내는 프레임 A와 프레임 B가 주어져있고 공간 상의 한 점을 $\mathbf{x}$라 하자. 여기서 $\mathbf{x}$는 3행 1열의 벡터이다. 이 점 $\mathbf{x}$를 프레임 A에서 본 좌표는 ${}^{A}\mathbf{x}$이고 프레임 B에서 본 좌표는 ${}^{B}\mathbf{x}$이다. 이 두 좌표의 관계식을 다음과 같이 회전변환 행렬 ${}^{A}_{B}\mathbf{R}$ 및 평행이동 벡터 ${}^{A}_{B}\mathbf{t}$를 사용하여 나타낼 수 있다.

$$^{A}\mathbf{x} =^{A}_{B} \mathbf{R}^{B}\mathbf{x} + ^{A}_{B}\mathbf{t} \tag{3.7}$$

3차원 공간 상의 점들의 집합인 하나의 평면 패치 $p = \{\mathbf{x}|\mathbf{n}^T\mathbf{x} = d\}$를 프레임 A, B에서 각각 보았을 때의 평면 패치는 $^{A}p = \{^{A}\mathbf{x}|^{A}\mathbf{n}^{TA}\mathbf{x} =^{A} d\}$, $^{B}p = \{^{B}\mathbf{x}|^{B}\mathbf{n}^{TB}\mathbf{x} =^{B} d\}$와 같이 나타낼 수 있다. 여기서 $\mathbf{n}$과 d는 각각 평면 패치 p의 법선벡터 및 원점과의 거리를 나타내고, 이를 프레임 A에서 본 $^{A}\mathbf{n}$ 및 ^{A}d는 프레임 A에서 본 평면 패치 ^{A}p 내부의 점들과 다음의 관계식을 가진다.

$$^{A}\mathbf{n}^{TA}\mathbf{x}_i = ^{A}d \tag{3.8}$$

마찬가지로 프레임 B에서 본 $^{B}\mathbf{n}$ 및 ^{B}d는 프레임 B에서 본 평면 패치 ^{B}p 내부의 점들과 다음의 관계식을 가진다.

$$^{B}\mathbf{n}^{TB}\mathbf{x}_i = ^{B}d \tag{3.9}$$

이 때, $^{A}\mathbf{x}_i =^{A}_{B} \mathbf{R}^{B}\mathbf{x}_i + ^{A}_{B}\mathbf{t}$가 성립하기 때문에 이 관계식을 식 3.8에 대입하면 다음과 같이 나타낼 수 있다.

$$^{A}\mathbf{n}^{T}(^{A}_{B}\mathbf{R}^{B}\mathbf{x}_i + ^{A}_{B}\mathbf{t}) = ^{A}d \tag{3.10}$$

이제 식 3.8과 식 3.10을 비교하면 $^{A}\mathbf{n}^{T}{}^{A}_{B}\mathbf{R} =^{B} \mathbf{n}^{T}$, $^{A}d - ^{A}\mathbf{n}^{T}{}^{A}_{B}\mathbf{t} =^{B} d$가 성립

함을 알 수 있고 양변에 transpose를 취하고 회전변환 행렬 ${}^A_B\mathbf{R}$이 ${}^A_B\mathbf{R}^T = {}^A_B\mathbf{R}^{-1}$ 임을 이용하여 식을 정리하면 다음과 같이 나타낼 수 있다.

$$ {}^A\mathbf{n} =^A_B \mathbf{R}\,{}^B\mathbf{n} \tag{3.11} $$

$$ {}^Ad =^B d + {}^A\mathbf{n}^T\,{}^A_B\mathbf{t} \tag{3.12} $$

여기서 확장하여 평면 패치 $p(\mathbf{n}, d, \boldsymbol{\mu}, \mathbf{S}, msd, k)$를 프레임 A와 B에서 각각 본 ${}^Ap({}^A\mathbf{n},{}^A\,d,{}^A\,\boldsymbol{\mu},{}^A\,\mathbf{S},{}^A\,msd,{}^A\,k)$ 및 ${}^Bp({}^B\mathbf{n},{}^B\,d,{}^B\,\boldsymbol{\mu},{}^B\,\mathbf{S},{}^B\,msd,{}^B\,k)$ 사이의 관계도 수학적으로 표기하자. 법선 벡터 $\mathbf{n}$와 원점과의 거리 d는 앞에서 다루었고 평면 패치를 구성하는 점들의 평균인 $\boldsymbol{\mu}$는 식 2.1에서 알 수 있듯이 각각의 점들 $\mathbf{x}_i$의 선형결합이기 때문에 똑같이 ${}^A\boldsymbol{\mu} =^A_B \mathbf{R}\,{}^B\boldsymbol{\mu} +^A_B \mathbf{t}$의 관계식을 만족한다. 공분산 행렬 S의 경우 식 2.2에서 ${}^A\mathbf{x} =^A_B \mathbf{R}\,{}^B\mathbf{x} +^A_B \mathbf{t}$, ${}^A\boldsymbol{\mu} =^A_B \mathbf{R}\,{}^B\boldsymbol{\mu} +^A_B \mathbf{t}$를 대입하면 ${}^A_B\mathbf{t}$가 소거되고 다음과 같이 정리할 수 있다.

$$ {}^A\mathbf{S} = {}^A_B\mathbf{R}\,{}^B\mathbf{S}\,{}^A_B\mathbf{R}^T \tag{3.13} $$

${}^Amsd$와 ${}^Bmsd$의 관계는 식 3.13 및 식 2.5를 통해 ${}^Amsd =^B msd$라는 것을 알 수 있다. 그리고 강체변환을 이용하여 평면 패치를 이동시켜도 점의 개수는 변하지 않기 때문에 ${}^Ak =^B k$ 역시 성립한다.

이제 일반적으로 프레임에서 평면 패치가 여러개 있을 경우를 고려한다. 원

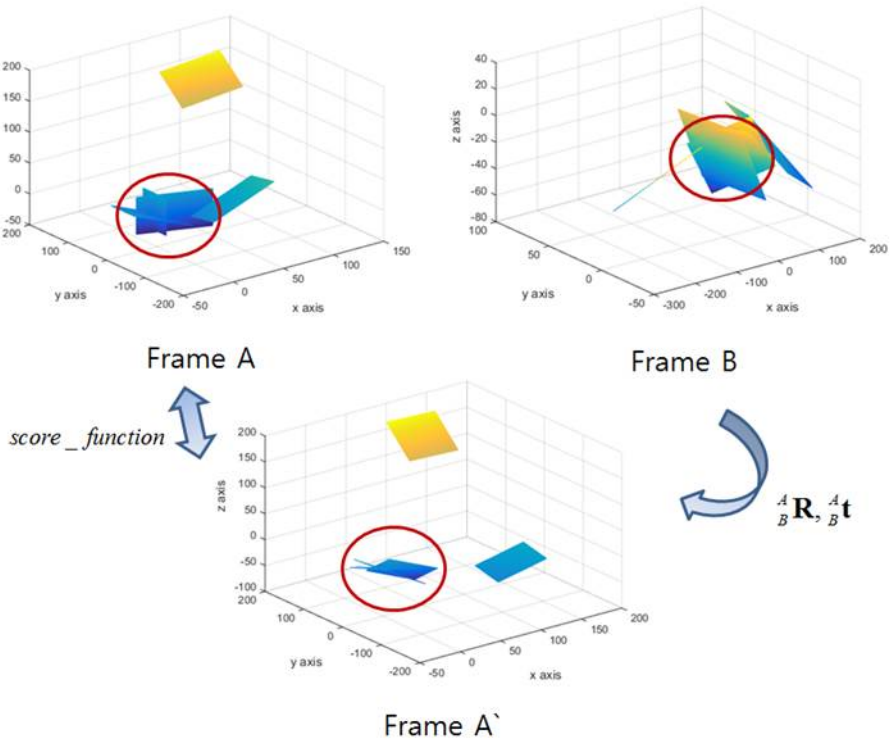


그림 3.6 Transformation of set of plane patches

점이 ${}^{B}O$인 프레임 B에서의 한 평면 패치를 ${}^{B}p_i({}^{B}\mathbf{n}_i, {}^{B}d_i, {}^{B}\boldsymbol{\mu}_i, {}^{B}\mathbf{S}_i, {}^{B}msd_i, {}^{B}k_i)$라 하자. 원점이 ${}^{A}O$인 프레임 A에서 본 프레임 B의 회전변환 행렬 및 평행이동 벡터가 각각 ${}^{A}_{B}\mathbf{R}$, ${}^{A}_{B}\mathbf{t}$로 주어져 있을 때, ${}^{B}p_i$를 변환시켜 얻은 프레임 A에서의 새로운 평면 ${}^{A}p_i$는 ${}^{A}p_i = \{\mathbf{x}|{}^{A}_{B}\mathbf{R}\mathbf{x} + {}^{A}_{B}\mathbf{t}, \mathbf{x} \in {}^{B}p_i\}$과 같이 표현될 수 있다. 이를 간단히 ${}^{A}p_i = T({}^{B}p_i, {}^{A}_{B}\mathbf{R}, {}^{A}_{B}\mathbf{t})$으로 나타낼 수 있다. 여기서 T는 한 프레임의 평면을 다른 프레임의 평면으로 변환시키는 함수이다. 이때 프레임 A상으로 변환된 프레임 B의 평면 패치인 ${}^{A'}p_i({}^{A'}\mathbf{n}_i, {}^{A'}d_i, {}^{A'}\boldsymbol{\mu}_i, {}^{A'}\mathbf{S}_i, {}^{A'}msd_i, {}^{A'}k_i)$와 프레임 B의

기존의 평면패치인 ${}^{B}p_i({}^{B}\mathbf{n}_i, {}^{B}d_i, {}^{B}\boldsymbol{\mu}_i, {}^{B}\mathbf{S}_i, {}^{B}msd_i, {}^{B}k_i)$는 각각 ${}^{A}\mathbf{n}_i = {}^{A}_{B}\mathbf{R}\,{}^{B}\mathbf{n}_i$, ${}^{A}\mathbf{n}_i{}^{T}\,{}^{A}_{B}\mathbf{t} = {}^{A}d_i - {}^{B}d_i$의 관계식을 만족한다. 또한 점들의 평균, 공분산 행렬은 각각 ${}^{A}\boldsymbol{\mu}_i = {}^{A}_{B}\mathbf{R}\,{}^{B}\boldsymbol{\mu}_i + {}^{A}_{B}\mathbf{t}$, ${}^{A}\mathbf{S} = {}^{A}_{B}\mathbf{R}\,{}^{B}\mathbf{S}\,{}^{A}_{B}\mathbf{R}^{T}$를 만족한다. 두 프레임의 평면이 완전히 같다고 가정하였기 때문에 ${}^{A}msd_i = {}^{B}msd_i$, ${}^{A}k_i = {}^{B}k_i$이다.

이를 확장하여 프레임 B에 있는 모든 평면패치들의 집합 ${}^{B}P = \{{}^{B}p_1, {}^{B}p_2, ..., {}^{B}p_{N_B}\}$에 ${}^{A}_{B}\mathbf{R}$, ${}^{A}_{B}\mathbf{t}$를 적용하여 프레임 A로 옮긴 새로운 평면 패치들의 집합 ${}^{A}P$는 ${}^{A}P = \{{}^{A}p_1, {}^{A}p_2, ..., {}^{A}p_{N_A}\}$와 같이 나타낼 수 있다. 이를 간단히 ${}^{A}P = T_{set}({}^{B}P, {}^{A}_{B}\mathbf{R}, {}^{A}_{B}\mathbf{t})$와 같이 나타낼 수 있다. 함수 T_{set}은 한 프레임에 있는 평면 패치들의 집합을 다른 프레임으로 변환시키는 함수이고 아래의 식을 만족한다.

$$T_{set}({}^{B}P, {}^{A}_{B}\mathbf{R}, {}^{A}_{B}\mathbf{t}) = \{T({}^{B}p_i, {}^{A}_{B}\mathbf{R}, {}^{A}_{B}\mathbf{t}) | {}^{B}p_i \in {}^{B}P\} \tag{3.14}$$

3.4 Mean Square Distance Function

우리가 해야 할 일은 프레임 A에서 본 프레임 B의 회전변환 행렬 ${}^{A}_{B}\mathbf{R}$및 평행이동 벡터 ${}^{A}_{B}\mathbf{t}$의 추정치를 구하는 것이다. 이 논문에서는 평면 패치를 기반으로 정합을 하기 때문에 프레임 B의 평면 패치들의 집합 ${}^{B}P$에 ${}^{A}_{B}\mathbf{R}$과 ${}^{A}_{B}\mathbf{t}$를 적용한 ${}^{A'}P = T_{set}({}^{B}P, {}^{A}_{B}\mathbf{R}, {}^{A}_{B}\mathbf{t})$와 프레임 A의 평면패치들의 집합인 ${}^{A}P$의 유사도를 판단하여 가장 유사도가 높은 ${}^{A}_{B}\mathbf{R}$과 ${}^{A}_{B}\mathbf{t}$를 구해야 한다.

두 개의 평면의 유사도를 판별하는데 가장 보편적으로 쓰이는 특성은 평면 패치의 법선벡터(normal vector) 및 원점과의 거리(distance to origin)이다. 두

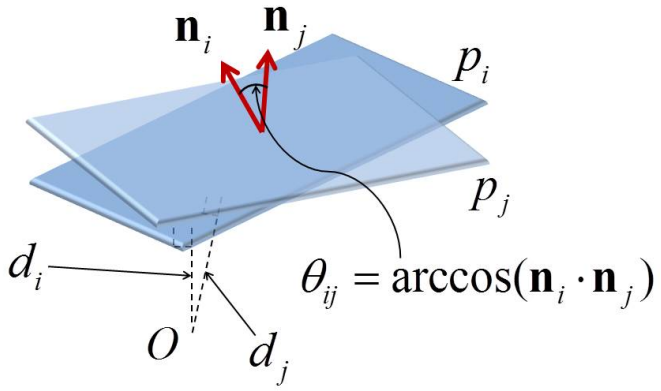


그림 3.7 Similarity evaluated by normal vector and distance to origin

평면의 법선벡터가 이루는 각도가 작을수록, 그리고 두 평면의 원점과의 거리의 차이가 작을수록 두 평면은 유사하다고 판단할 수 있다. 그림 3.7에서와 같이 두 개의 평면 패치 p_i, p_j가 있고 각각의 법선벡터 및 원점과의 거리를 $\mathbf{n}_i$, d_i 및 $\mathbf{n}_j$, d_j라 하자. 그리고 두 법선벡터가 이루는 각도를 θ_{ij}라 할 때, $\mathbf{n}_i$와 $\mathbf{n}_j$가 단위벡터이면 두 벡터를 내적한 값이 바로 $\cos\theta_{ij}$와 같다. 따라서 유사도를 비교하고자 하는 두 평면 패치의 법선벡터를 내적한 값에 arccos 값을 취하여 두 평면이 이루는 각도를 구할 수 있다. 이렇게 구한 $\cos\theta_{ij}$와 $d_i - d_j$가 지정한 threshold 값보다 작은 경우 두 평면이 유사하다고 판단할 수 있다.

그러나 $\arccos x$함수의 기울기는 $-\frac{1}{\sqrt{1-x^2}}$와 같고 유사한 두 평면의 법선벡터를 서로 내적한 값은 1에 가깝기 때문에 그림 3.8와 그림 3.9에서 볼 수 있듯이

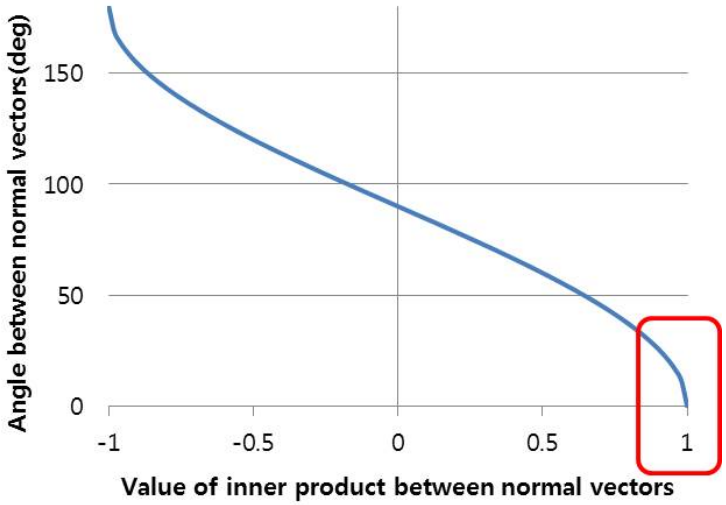


그림 3.8 graph of arccos function

이 값에 대한 arccos 값이 조금만 변해도 arccos값을 적용한 사이각이 급격하게 변하게 된다. 따라서 약간의 noise에도 사이각이 많이 왜곡되어 유사도를 판별할 때 잘못된 결과를 초래할 수 있다.

이러한 단점을 보완하기 위해 두 평면패치 ${}^{A}P$와 ${}^{A'}P$의 유사도를 판단하는 기준으로 '두 평면패치들의 집합 간의 거리제곱평균'이라는 개념을 사용하고 이 값이 작을수록 두 평면패치들의 집합이 유사하다고 판단한다. '두 평면패치들의 집합 간의 거리제곱평균'을 정의하기 위하여 먼저 같은 프레임 상에 있는 두 평면패치 ${}^{A}p_i$, ${}^{A'}p_j$를 생각한다. 이 때, 평면패치 ${}^{A}p_i$를 기준으로 한 평면패치 ${}^{A'}p_j$와의 거리제곱평균을 다음 식과 같이 정의한다.

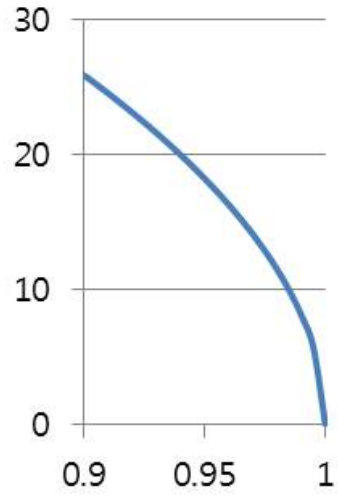


그림 3.9 graph of arccos function

$$D_{plane}({}^{A'}p_j|{}^{A}p_i) \equiv \sum_{\mathbf{x}\in {}^{A'}p_j} \frac{(\mathbf{x}^{T\,A}\mathbf{n}_i - {}^{A}d_i)^2}{{}^{A'}k_j} \tag{3.15}$$

식 3.15에서 ${}^{A}p_i$는 기준이 되는 기준 평면패치이고 ${}^{A'}p_j$는 거리를 구할 점들을 가지고 있는 비교 평면패치이다. 즉 기준 평면패치의 평면과 비교 평면패치의 점들의 거리제곱평균을 두 평면 패치 사이의 거리제곱평균으로 정의한 것이다. 이를 정리하여 다음과 같이 변형할 수 있다.

$$D_{plane}({}^{A'}p_j|{}^{A}p_i) = \frac{{}^{A'}N_j}{{}^{A}N_i}{}^{A}\mathbf{n}_i^T({}^{A'}\mathbf{S}_j + {}^{A'}\boldsymbol{\mu}_j{}^{A'}\boldsymbol{\mu}_j{}^T){}^{A}\mathbf{n}_i - 2{}^{A}d_i\frac{{}^{A'}N_j}{{}^{A}N_i}({}^{A}\mathbf{n}_i{}^{T\,A'}\boldsymbol{\mu}_j) + {}^{A}d_i{}^2 \tag{3.16}$$

하나의 평면 패치 ${}^{A'}p_j$와 한 평면 패치들의 집합 ${}^{A}P = \{{}^{A}p_1, {}^{A}p_2, ..., {}^{A}p_{N_A}\}$

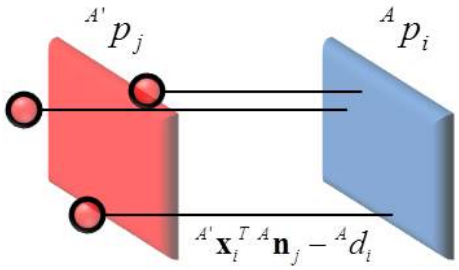


그림 3.10 Mean square distance between two plane patches

사이의 거리제곱평균은 ${}^{A'}p_j$와 ${}^{A}P$의 한 원소의 거리제곱평균 중 최소값으로 정의한다.

$$D_{plane}({}^{A'}p_j|{}^{A}P) \equiv \min_{i\in\{1,2,...,N_A\}} D_{plane}({}^{A'}p_j|{}^{A}p_i) \tag{3.17}$$

이제 평면패치들의 집합 ${}^{A}P$를 기준으로 한 평면패치들의 집합 ${}^{A'}P = \{{}^{A'}p_1, {}^{A'}p_2, ..., {}^{A'}p_{N_{A'}}\}$와의 거리는 ${}^{A}P$를 기준으로 한 ${}^{A'}P$의 각 원소와의 거리제곱평균의 평균값으로 정의한다.

$$D_{plane}({}^{A'}P|{}^{A}P) \equiv \sum_{j=1}^{N_{A'}} \frac{D_{plane}({}^{A'}p_j|{}^{A}P)}{N_{A'}} \tag{3.18}$$

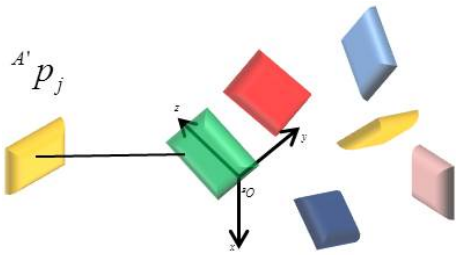


그림 3.11 Mean square distance between a plane patch and a set of plane patches

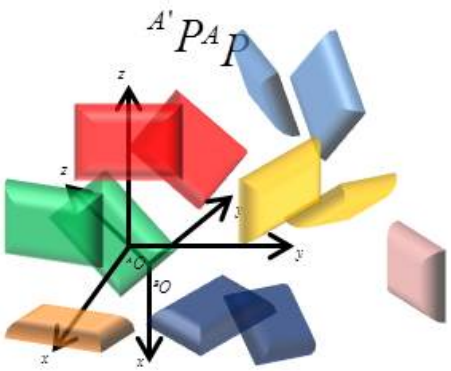


그림 3.12 Mean square distance between two sets of plane patches

3.5 Selection of Rigid Transformation

k번의 반복작업을 하며 각각의 step에 대하여 ${}^A_B\mathbf{R}$ 및 ${}^A_B\mathbf{t}$를 구하게 되는데, 이 중에 두 프레임 A와 B를 가장 잘 매칭시키는 회전변환 행렬과 평행이동 벡터를 구해야 한다. 이는 프레임 B에 강체변환을 적용한 프레임 A'과 기존의 프레임 A를 가장 유사하게 만드는 ${}^A_B\mathbf{R}$과 ${}^A_B\mathbf{t}$를 구하는 것이고 위에서 정의한 평면 패치들의 집합 사이의 평균제곱거리의 개념 및 변환함수 T를 이용하여 표현할 수 있다. 즉, k번의 반복작업 도중 $D_{plane}({}^{A'}P|{}^AP)$의 값을 최소로 하는 ${}^A_B\mathbf{R}^*$과 ${}^A_B\mathbf{t}^*$를 1차 회전변환 행렬 ${}^A_B\mathbf{R}_1 = {}^A_B\mathbf{R}^*$ 및 평행이동 벡터 ${}^A_B\mathbf{t}_1 = {}^A_B\mathbf{t}^*$ 로 반환한다.

$$ {}^A_B\mathbf{R}^*, {}^A_B\mathbf{t}^* = \underset{{}^A_B\mathbf{R}, {}^A_B\mathbf{t}}{\operatorname{argmin}}[D_{plane}\{T_{set}({}^BP, {}^A_B\mathbf{R}, {}^A_B\mathbf{t})|{}^AP\}] \tag{3.19} $$

3.6 Optimization of Translation Vector

반복작업을 마친 후 구해진 ${}^{A}_{B}\mathbf{R}_1$ 과 ${}^{A}_{B}\mathbf{t}_1$, 그리고 $D_{plane}({}^{A'}P|{}^{A}P)$에 대하여 최적화 과정이 추가로 필요하다. $D_{plane}({}^{A'}P|{}^{A}P)$은 L번의 iteration 중 가장 작은 값으로 선택이 된 것일 뿐 고정된 ${}^{A}_{B}\mathbf{R}_1$에 대하여 최소화가 되어있지 않은 상태이다. 따라서 ${}^{A}_{B}\mathbf{t}_1$에 보정 평행이동벡터 $\mathbf{t}_{cal}$을 더하여 최종 회전이동 행렬 ${}^{A}_{B}\mathbf{R}_{final}$ 및 평행이동 벡터 ${}^{A}_{B}\mathbf{t}_{final}$ 를 구한다. $\mathbf{t}_{cal}$을 ${}^{A'}P$에 적용한 후 $D_{plane}({}^{A'}P|{}^{A}P)$의 값이 최소가 되어야 하기 때문에 $\mathbf{t}_{cal}$의 값을 다음과 같이 표현할 수 있다.

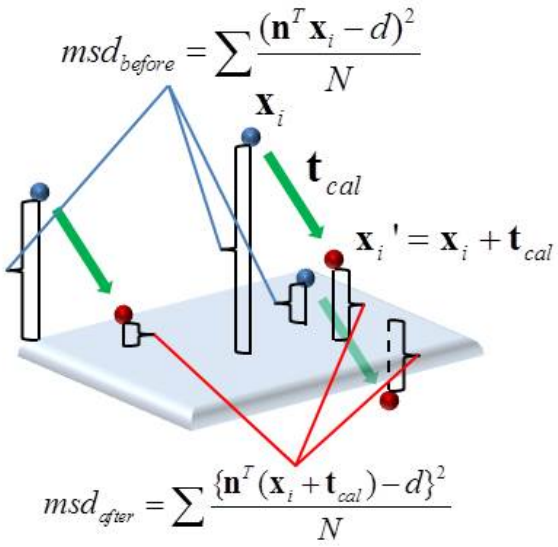


그림 3.13 Calibration Translation Vector $\mathbf{t}_{opt}$ for single plane patch

$$\mathbf{t}_{cal} = \underset{\mathbf{t}}{\operatorname{argmin}}[D_{plane}\{T_{set}({}^{A'}P, \mathbf{I}_{3\times3}, \mathbf{t})|{}^{A}P\}] \tag{3.20}$$

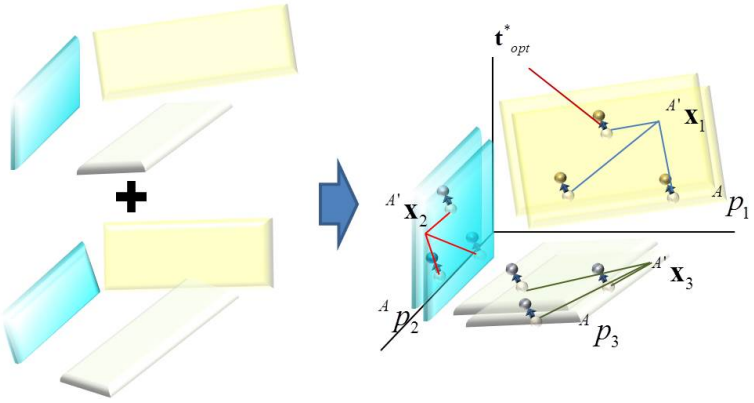


그림 3.14 Calibration Translation Vector $\mathbf{t}_{opt}$ in two frames

$D_{plane}\{T_{set}({}^{A'}P, \mathbf{I}_{3\times3}, \mathbf{t})|{}^{A}P\}$는 $D_{plane}({}^{A'}P|{}^{A}P)$이 고정되어 있을 때 오직 $\mathbf{t}$에 대하여 quadratic한 형태이기 때문에 이 값을 최소화 시키는 closed form의 해가 존재하며 다음 식과 같이 나타낼 수 있다.

$$\mathbf{t}^*_{opt} = \frac{\sum_{k=1}^{N_c} \frac{{}^{B}N_k}{{}^{A}N_k}({}^{A}d_k\,{}^{A}\mathbf{n}_k - {}^{A}\mathbf{n}_k\,{}^{A}\mathbf{n}_k{}^{T}\,{}^{A'}\boldsymbol{\mu}_k)}{\sum_{k=1}^{N_c} \frac{{}^{B}N_k}{{}^{A}N_k}\,{}^{A}\mathbf{n}_k\,{}^{A}\mathbf{n}_k{}^{T}} \tag{3.21}$$

따라서 $\mathbf{t}_{cal} = \mathbf{t}^*_{opt}$이고 최종적으로 ${}^{A}_{B}\mathbf{R}_{final} = {}^{A}_{B}\mathbf{R}^*$, ${}^{A}_{B}\mathbf{t}_{final} = {}^{A}_{B}\mathbf{t}^* + \mathbf{t}_{cal}$ 과 같이 ${}^{A}_{B}\mathbf{R}_{final}$ 과 ${}^{A}_{B}\mathbf{t}_{final}$ 를 구하면 정합이 완료된다.

제 4 장 Simulations

제안한 알고리즘의 성능을 검증하기 위하여 Matlab을 기반으로 시뮬레이션을 수행하였다. 이 기법의 핵심인 임의 대응쌍의 타당성을 보이기 위하여 임의 대응쌍을 사용하여 반복적으로 강체변환을 구하는 과정과 모든 경우를 고려하여 강체변환을 구하는 과정을 구현하여 수행시간 및 정합의 성공률, 그리고 성공하였을 때 회전변환 및 평행이동 변환의 오차를 측정하여 비교하였다. 그 다음에는 기존에 연구되어있는 평면 기반 정합기법인 MUMC 및 PRRUS로 정합을 수행하였을 때의 수행시간, 정합의 성공률, 성공 시의 오차를 측정하여 제안한 기법과 비교하였다.

4.1 Preconditions for the Simulation

제안한 기법을 시뮬레이션에 적용하기에 앞서 필요한 전제조건이 있다. 프레임 A와 프레임 B를 나타내는 평면 패치들의 집합 ${}^{A}P = \{{}^{A}p_1, {}^{A}p_2, ..., {}^{A}p_{N_A}\}$과 ${}^{B}P = \{{}^{B}p_1, {}^{B}p_2, ..., {}^{B}p_{N_B}\}$이 있고 올바른 대응쌍 $U(A,B) = \{({}^{A}p_{c_i}, {}^{B}p_{c_i}) | i = 1, 2, ..., N_c\}$에 대하여 $N_A \geq 3$, $N_B \geq 3$, 그리고 $N_c \geq 3$을 만족해야 한다. 평면 패치들의 대응쌍을 정하여 그를 바탕으로 회전변환 행렬과 평행이동 벡터가 유일하게 결정되려면 최소한 세 쌍 이상의 대응관계가 필요하다. 그렇지 않으면 하나의 대응쌍이 주어졌을 때에도 무수히 많은 회전변환 행렬과 평행이동 벡터가 해가 법선벡터와 원점과의 거리에 대한 방정식을 만족하는 해가 된다. 그렇기 때문

에 시뮬레이션을 수행할 때 항상 이 세 변수들은 모두 3 이상의 값으로 지정하였다. 또한 대응쌍으로 주어진 평면 중 평행한 평면들이 있을 경우에 역시 평행이동 벡터를 구하는 과정에서 해가 유일하게 결정되지 않기 때문에 시뮬레이션을 수행할 때 완전히 평행인 평면이 없도록 평면 패치를 생성하였다.

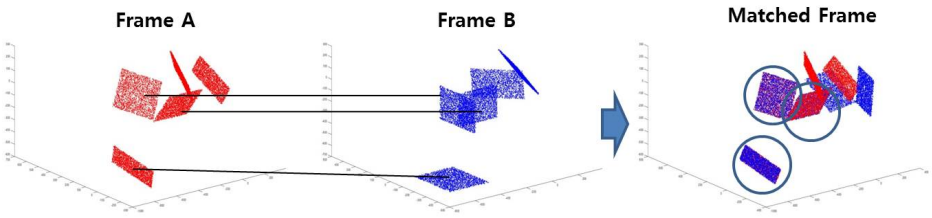


그림 4.1 Example of simulation

4.2 Validity of Random Iteration

이 절에서는 제안한 알고리즘에서 강체변환을 구하기 위하여 반복적으로 평면 패치들의 임의 대응쌍을 지정하는 과정의 타당성에 대하여 설명한다. 반복작업을 수행할 때와 모든 대응쌍을 고려하는 경우 각각에 대하여 경우의 수와 확률을 수학적으로 계산하여 반복작업의 회수와 프레임 내부의 전체 평면 패치의 개수 및 대응쌍의 개수에 따라 정합 성공률이 어떻게 표현되는지를 나타내었다. 그 다음 Matlab에서 정합을 하고자 하는 두 프레임 A, B를 대표하는 평면 패치들의 집합 ${}^{A}P$ 및 ${}^{B}P$를 생성한 후 반복적으로 대응쌍을 구할 때와 모든 대응쌍의 경우의 수를 고려할 때 각각에 대한 정합의 성공률 및 수행시간을 측정하였다. 제안한 기법 대로 정합을 하는 경우 ${}^{A}P$과 ${}^{B}P$에서 임의로 N개의 서로 다른 평면 패치를

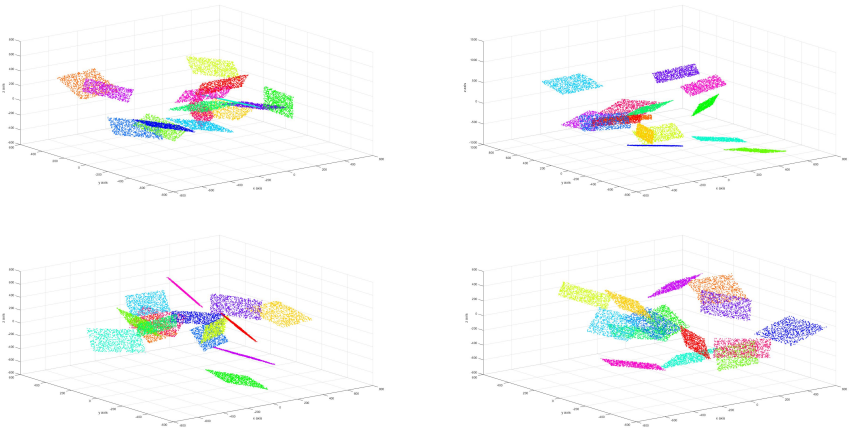


그림 4.2 Example of frame A

각각 추출하는 과정을 정해진 횟수 만큼 반복하기 때문에 이 때 수행되는 작업의 빈도수는 반복회수 k와 같다. 반면, 모든 대응쌍을 고려하는 경우 전체 대응쌍의 개수는 다음과 같이 구할 수 있다. 프레임 A에서 서로 다른 평면 패치 N개를 뽑는 경우의 수는 $\binom{N_A}{N}$이고 그 다음 프레임 B에서 서로 다른 평면 패치 N개를 뽑을 때에는 순서가 고려되기 때문에 그 가지수가 $\binom{N_B}{N} \times N!$이 된다. 따라서 대응쌍의 전체 개수는 $\binom{N_A}{N} \times \binom{N_B}{N} \times N!$와 같다. 최소한 세 쌍 이상의 대응쌍이 있어야 회전변환 행렬과 평행이동 벡터의 값이 유일하게 결정되기 때문에 $N \geq 3$이고 N_A와 N_B가 모두 5 이상일 경우 $\binom{N_A}{N} \times \binom{N_B}{N} \times N! \geq \binom{N_A}{3} \times \binom{N_B}{3} \times 3!$이 되고 이는 N_A와 N_B가 큰 경우에는 약 $({N_A}^3 \times {N_B}^3)/6$과 근사적으로 같다.

한편, 임의 대응쌍을 사용하여 정합을 할 때 성공할 확률을 수학적으로 기술하면 다음과 같다. 프레임 A와 B에 각각 N_A개, N_B개의 평면 패치가 있고 이 중에 올바른 대응관계에 속하는 평면 패치들이 $U(A,B) = \{({}^{A}p_{c_i}, {}^{B}p_{c_i}) | i = 1, 2, ..., N_c\}$

로 주어져 있다면 각 프레임에서 inlier의 개수는 N_c라 할 수 있다. 따라서 임의 대응쌍과 그에 해당하는 강체변환을 구하는 작업을 한 번 수행할 때 inlier안에 있는 대응쌍을 올바르게 선택하여 성공적으로 강체변환을 구할 확률을 수학적으로 나타낼 수 있다. 강체변환을 성공적으로 구한다는 것은 임의로 고른 대응쌍이 $U(A,B)$의 원소 중 3개가 선택되었다는 의미이고 그 경우의 수는 $\binom{N_c}{N}$과 같다. 이를 정리하면 작업을 한 번 수행할 때 올바른 강체변환을 구할 확률 α은 다음과 같다.

$$\alpha = \frac{\frac{N_c!}{(N_c-N)!N!}}{\frac{N_A!}{(N_A-N)!N!} \times \frac{N_B!}{(N_B-N)!N!} \times N!} \tag{4.1}$$

따라서 k번의 반복작업을 수행할 때 정합에 성공할 확률은 1에서 k번의 반복작업 중 정합에 한 번도 성공하지 못할 확률을 빼면 되고, 이는 다음과 같이 나타낼 수 있다.

$$\eta_0 = 1 - (1-\alpha)^k \tag{4.2}$$

제안한 수식을 바탕으로 Matlab에서 시뮬레이션을 수행하였다. 시뮬레이션 환경은 다음과 같이 구성하였다. 두 프레임을 나타내는 평면 패치들의 집합 ${}^AP$ 및 ${}^BP$를 각각 생성하고 올바른 대응쌍 $U(A,B)$ 및 강체변환 ${}^A_B\mathbf{R}$ 과 ${}^A_B\mathbf{t}$의 Ground Truth 값을 지정하였다. Ground Truth를 만들기 위하여 ${}^A_B\mathbf{R}$은 roll, pitch, yaw의 값에 따라 변화하도록 지정하였다. 대응쌍 $U(A,B)$에 속하는 원소들 중 프레임

B에 속하는 평면 패치들 ${}^{B}p_{c_i}$들을 먼저 생성하였고 ${}^{A}_{B}\mathbf{R}$ 과 ${}^{A}_{B}\mathbf{t}$의 Ground Truth 값을 적용하여 $U(A,B)$에 속하는 원소들 중 프레임 A에 속하는 평면 패치들 ${}^{A}p_{c_i}$를 만들었다. 이 때 행렬 ${}^{A}_{B}\mathbf{R}$을 곱하고 ${}^{A}_{B}\mathbf{t}$을 더한 다음 Gaussian noise도 추가하여 $U(A,B)$를 만들었다. 그리고 프레임 A와 B에서 나머지 outlier에 해당하는 평면 패치들을 각각 $N_A - N_c$개, $N_B - N_c$개 생성하여 전체 시뮬레이션 환경을 만들었다.

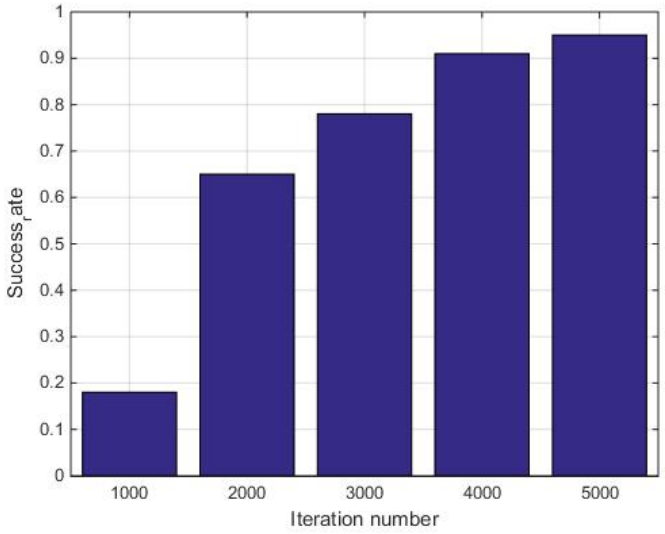


그림 4.3 Success rate of random iteration

	Random Iteration	Considering all the cases
Success rate(%)	98.7	99.5
Rotation error(deg)	1.55	1.54
Translation error(cm)	0.012	0.015
Execution time(s)	0.387	0.467

표 4.1 Comparison between random iteration and considering all the cases. k=4000

고정된 시뮬레이션 환경에서 임의 대응쌍을 뽑는 과정의 반복회수 k를 1000부터 5000까지 변화시키며 정합의 성공률 및 평균 수행시간을 측정하였고 같은 환경에서 모든 대응쌍을 고려하여 정합을 한 경우도 같이 비교를 하였고 그 결과는 그림 4.3 및 그림 4.4과 같았다. roll, pitch, yaw의 값이 각각 10도, 20도, 30도인 회전변환에 대응하는 행렬, 평행이동 벡터의 참값은 ${}^A_B\mathbf{t} = [20\ 15\ 10]^T$으로 지정한 경우에 대하여 임의 대응쌍 . $N_A = N_B = 5$, $N_c = 3$일 때 프레임 A의 평면 패치들 ${}^AP = {}^Ap_1, {}^Ap_2, ..., {}^Ap_{N_A}$을 생성한 후, 그 중 ${}^Ap_2, {}^Ap_3, {}^Ap_4$에 변환 $T(;{}^A_B\mathbf{R}, {}^A_B\mathbf{t})$를 적용하고 Gaussian Noise를 추가하여 프레임 B의 평면패치들 ${}^BP = {}^Bp_1, {}^Bp_2, ..., {}^Bp_{N_B}$ 중 일부 ${}^Bp_2, {}^Bp_3, {}^Bp_4$를 생성하였다. 그 다음 프레임 B에 2개의 평면패치 ${}^Bp_1, {}^Bp_5$를 추가하여 프레임 B를 만들었다. 제안한 알고리

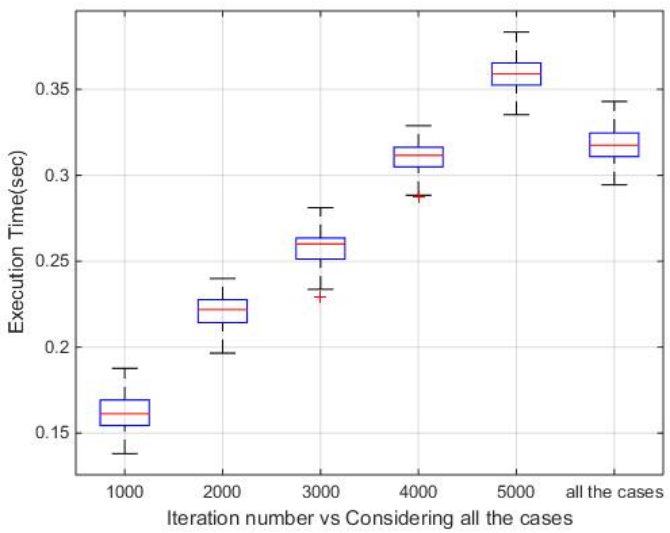


그림 4.4 Execution time of random iteration

즘을 적용시킨 결과 $U(A,B) = \{({}^{A}p_2, {}^{B}p_2), ({}^{A}p_3, {}^{B}p_3), ({}^{A}p_4, {}^{B}p_4)\}$으로 올바른 대응쌍을 찾아내었고 회전변환 행렬과 평행이동 벡터의 추정값 ${}^{A}_{B}\mathbf{R}_{final}$과 ${}^{A}_{B}\mathbf{t}_{final}$을 구한 결과 각각의 평균 오차는 표 4.1과 같았다.

4.3 Simulation Results

반복적으로 임의 대응쌍을 선택하는 본 논문의 기법을 기존의 평면 정합 알고리즘인 MUMC 및 PRRUS 기법과 비교를 하였다. N_A, N_B 및 N_c를 변화시킨 다양한 환경에서 세 알고리즘으로 정합을 수행하였다. 그 결과 프레임 내의 평면 패치의 개수, 즉 N_A와 N_B가 클수록 제안한 기법의 수행시간이 더 작음을 확인할 수 있었다.

$N_A = N_B = 30$일 때 제안한 기법의 평균 수행시간은 1.16초로 PRRUS 기법의 평균 수행시간인 1.76초 보다 34.1% 감소하였고, $N_A = N_B = 50$일 때는 제안한 기법의 평균 수행시간이 약 1.30초로 MUMC 기법의 평균 수행시간인 2.24초 보다 42.0% 짧았고 PRRUS 기법의 평균 수행시간인 2.19초 보다 40.6% 짧았다.

또한 프레임 내의 outlier의 비율, 즉 $1 - \frac{N_c}{N_A}$와 $1 - \frac{N_c}{N_B}$가 높을수록 제안한 알고리즘의 정합 성공률이 우세함을 확인할 수 있었다. Outlier의 비율이 50%일 때 제안한 기법의 정합 평균 성공률은 68%로 MUMC 기법보다 17.2% 높았고 PRRUS 기법보다 36% 높았다. Outlier의 비율이 70%일 때는 제안한 기법의 정합 성공률이 44%로 MUMC 기법보다 63.0%, PRRUS 기법보다 41.9% 높았다.

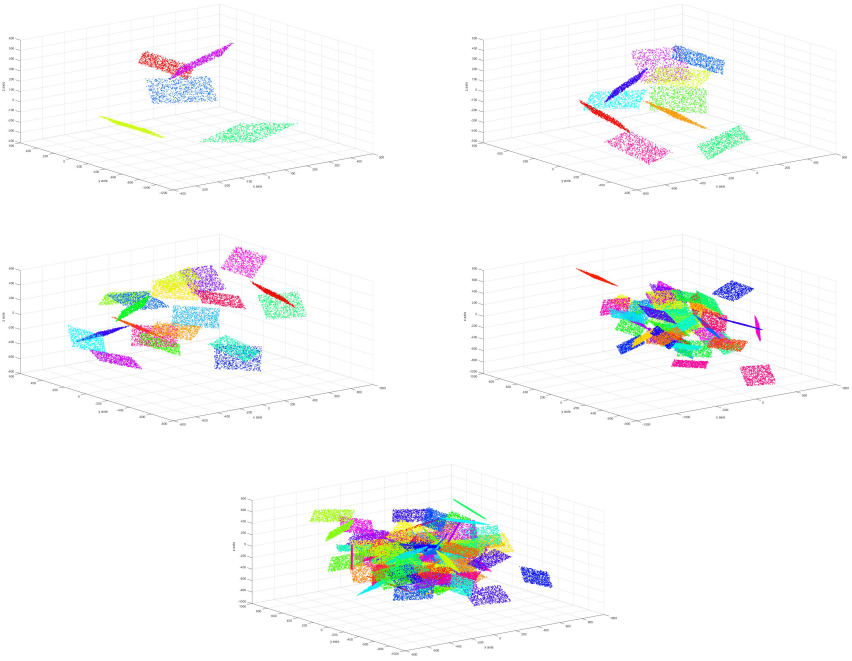


그림 4.5 Common Frame number 5-100

	MUMC	PRRUS	Proposed Method
Success rate(%)	27	31	44
Rotation error(deg)	6.54	5.81	6.31
Translation error(cm)	1.3	1.13	1.23
Execution time(s)	2.63	2.21	1.49

표 4.2 Frame number 50, Outlier rate 70%

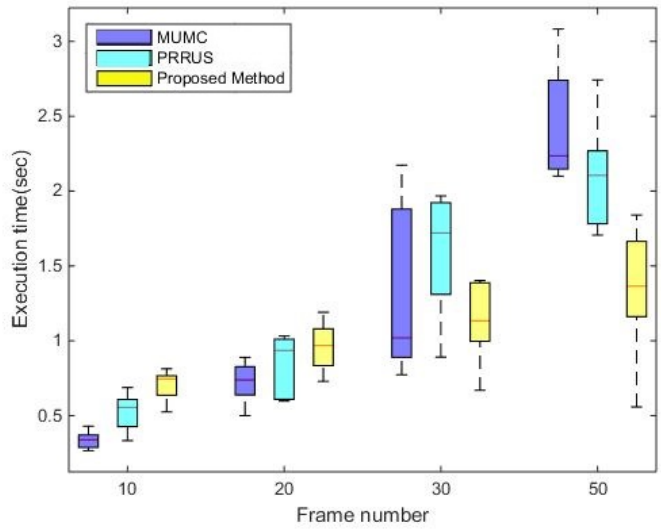


그림 4.6 Simulation Results-Execution time

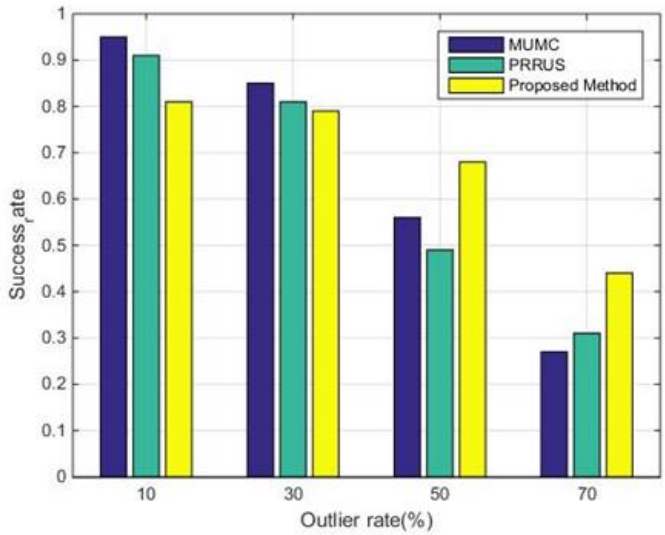


그림 4.7 Simulation Results-Success rate

제 5 장 Real Experiments

이 절에서는 실제 환경에서 본 논문의 기법을 바탕으로 평면 정합을 하였을 때의 결과에 대하여 다룬다. 직접 얻은 이미지 파일을 기반으로 평면 패치를 추출하여 정합을 하였고 제안한 기법의 타당성을 증명하기 위하여 MUMC, PRRUS, 그리고 제안한 기법 총 세 가지 방법으로 평면을 정합하였다.

5.1 Environments for the Experiments

실제 환경에서의 실험은 서울대학교 신공학관 716호에서 진행되었으며(그림 5.1) RGB-D 카메라 Xtion Pro를 사용하여 RBG 이미지 및 Depth 이미지를 얻었다. 그림 5.2과 같이 카메라를 약 5도의 간격으로 돌려가며 총 316개의 이미지 파일을 추출하였고 이렇게 얻은 이미지를 3차원 Point cloud로 변환을 하였다.

그림 5.1 Environment of real experiments

5.2 Extraction of plane patches from point cloud

RGB-D 이미지를 point cloud로 변환한 후에는 여기서 평면을 추출하는 과정이 필요하다. 평면을 추출하는 기법에는 Region growing [24], Robust PCA [25] 등의 기법이 있지만 이 논문에서는Microsoft사가 제공하는 Opencv Library의 함수를 이용하여 평면 패치를 추출하였다. Point cloud에서 하나의 평면 패치에 속하는 점들을 추출하고 남은 Point cloud에서 다시 평면 패치를 추출하는 과정을 재귀적으로 반복하였다. 이 과정의 종료조건은 추출된 평면 패치를 구성하고 있는 점의 개수가 300개 이하일 때 혹은 더이상 추출할 수 있는 평면 패치가 없을 때로 지정하였다.

그림 5.2 3D point cloud extracted from the RGB-D image

5.3 Results of Real Experiments

추출한 평면 패치들을 기반으로 MUMC, PRRUS와 제안한 기법 총 3가지 방법으로 평면 패치들을 정합을 수행하였고 그 결과는 표 5.3 및 그림 5.3과 같았다. 제안한 기법의 정합 성공률은 평균 53%로 MUMC에 비하여 3.9%, PRRUS에 비하여 10.4% 증가하였다. 그리고 수행시간은 1.18초로 MUMC에 비하여 52.6%, PRRUS에 비하여 43.3% 감소하였다.

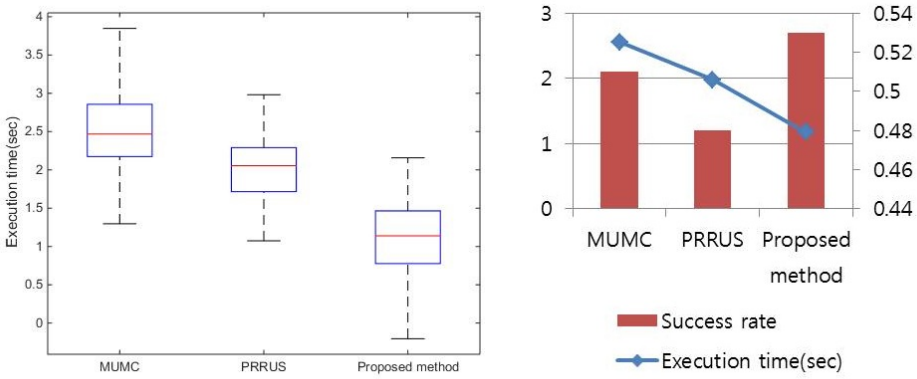


그림 5.3 Results of real experiments-execution time and success rate

	MUMC	PRRUS	Proposed Method
Success rate(%)	51	48	53
Execution time(s)	2.56	1.98	1.18

표 5.1 Success rate and Execution time

제 6 장 Conclusion

이 논문에서는 3차원 공간에서 적용할 수 있는 새로운 정합 기법에 대하여 다루었다. 이 기법은 평면 패치를 기반으로 정합을 하는데, 기존 기법들의 문제인 대응쌍 문제를 해결하기 위하여 새로운 방법을 시도하였다. 두 프레임에서 각각 임의로 평면을 선택하여 대응쌍을 만든 다음 그에 해당하는 강체변환을 구하여 한 프레임을 나머지 한 프레임에 유사한 프레임으로 변환한다. 변환된 프레임과 원래의 프레임의 유사도를 측정하는 작업을 반복적으로 수행한 후 가장 유사하게 두 프레임을 정합시키는 대응쌍 및 강체변환을 최종 결과값으로 반환한다. 프레임간 유사도를 측정하는 기준으로 기존에 있던 법선벡터의 사이각, 원점과의 거리의 차이 이외에도 평면패치 사이의 평균제곱거리를 정의하여 정합을 수행하였다. 고정된 강체변환에 대하여 평균제곱거리를 최소화 시키는 평행이동벡터의 보정값을 찾은 후 이를 더하여 평행이동벡터를 보정하여 최종적으로 정합을 완료한다.

제안한 기법에서 임의 대응쌍의 타당성을 입증하기 위하여 시뮬레이션을 통해 모든 경우를 고려할 때에 비하여 정확성이 크게 떨어지지 않으며 수행시간을 단축시킬 수 있다는 사실을 확인하였다. 또한 기존에 연구되어있던 다른 평면 정합 기법과 정합 성공률, 수행시간, 정합의 정확도를 측정하여 제안한 기법의 타당성을 증명하였다. 실제 환경에서의 실험 또한 기존의 두 기법에 비하여 제안한 기법의 성공률 및 수행시간이 개선됨을 확인할 수 있었다.

참고문헌

[1] T. Lemaire, C. Berger, I.K. Jung and S. Lacroix, "Vision-Based SLAM: Stereo and Monocular Approaches," *International Journal of Computer Vision.*, vol. 74, no.3, pp. 343-364, 2007.

[2] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, vol. 14, no.2, pp. 239-256, 1992.

[3] L. Armesto, J. Minguezand, and L. Montesano, "A Generalization of the Metric-Based Iterative Closest Point Technique for 3D Scan Matching," in *2010 IEEE International Conference on Robotics and Automation*, Anchorage, USA, 2010.

[4] J. P. Saarinen, H. Andreasson, T. Stoyanovand A. J. Lilienthal, "3D Normal Distributions Transform Occupancy Maps: An Efficient Representation for Mapping in Dynamic Environments," *The International Journal of Robotics Research*, vol. 32, no.14, pp. 1627-1644, 2013.

[5] M. Mohamad, D. Rappaport and M. Greenspan, "Generalized 4-points congruent sets for 3D registration," in *3D Vision*, 2014 2nd International

Conference, Dec. 2014.

[6] W.Y. Jeong and K.M. Lee, "Visual SLAM with Line and Corner Features," in *International Conference on Intelligent Robots and Systems*, 2006.

[7] F. Bosche, "Plane-based Registration of Construction Laser Scans with 3D/4D Building Models," *Advanced Engineering Informatics*, vol. 26, no.1, pp. 90-102, 2012.

[8] Z. Lou and T. Gevers, "Image alignment by piecewise planar region matching," *IEEE Transactions on Multimedia*, vol. 16, no.7, pp. 2052-2061, 2014.

[9] B. He, Z. Lin and Y.F. Li, "An Automatic Registration Algorithm for the Scattered Point Clouds based on the Curvature Feature," *Optics & Laser Technology*, vol. 46, pp. 53-60, 2013.

[10] K. Pathak, A. Birk, N. Vaskevicius and J. Poppinga, "Fast Registration Based on Noisy Planes With Unknown Correspondences for 3-D Mapping," *IEEE Transactions on robotics*, vol. 26, no.3, pp. 424-441, 2010.

[11] K. Pathak, N. Vaskevicius, J. Poppinga, M. Pfingsthorn, S. Schwertfeger and A. Birk, "Fast 3D Mapping by Matching Planes Extracted from Range Sensor Point-Clouds," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[12] J. Weingarten and R. Siegwart, "3D SLAM using planar segments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2006.

[13] E. Eade and T. Drummond, "Monocular SLAM as a Graph of Coalesced Observations," in *IEEE 11th International Conference on Computer Vision*, Oct 2007.

[14] R. Raguram, J. M. Frahm, and M. Possefeys, "A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus," in *10th European Conference on Computer Vision*, Marseille, France, Oct 2008. pp. 500-513.

[15] D. Fontanelli, L. Ricciato and S. Soatto, "A Fast RANSAC-Based Registration Algorithm for Accurate Localization in Unknown Environments using LIDAR Measurements," in *IEEE International Conference on Automation Science and Engineering*, Sept 2007. pp. 597-602.

[16] J. Civera, O. G. Grasa, A. J. Davison and J. M. M. Montiel, "1-Point RANSAC for extended Kalman filtering: Application to real-time structure from motion and visual odometry," *Journal of Field Robotics*, vol. 27, no.5, pp. 609-631, Oct 2010.

[17] E. Serradell, P. Glowacki, J. Kybic, F. Moreno-Noguer and P. Fua, "Robust non-rigid registration of 2D and 3D graphs," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2012. pp. 996-1003.

[18] S.Y. Park and M. Subbarao, "An accurate and fast point-to-plane registration technique," *Pattern Recognition Letters*, vol. 24, no.16, pp. 2967-2976, Dec 2003.

[19] T. Stoyanov, M. Magnusson, H. Andreasson and A. J. Lilienthal, "Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations," *The International Journal of Robotics Research*, vol. 31, no.12, pp. 1377-1393, Oct 2012.

[20] R. Hulik, V. Beran, M. Spanel, P. Krsek and P. Smrz, "Fast and Accurate Plane Segmentation in Depth Maps for Indoor Scenes," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, Algarve, Portugal, Oct 2012. pp. 1665-1670.

[21] Z. Wang, H. Liu, Y. Qian and T. Xu, "Real-Time Plane Segmentation and Obstacle Detection of 3D Point Clouds for Indoor Scenes," in *European Conference on Computer Vision*, Oct 2012. pp. 22-31.

[22] B. R. Fortenbury and G. Guerra-Filho, "Robust 2D/3D Calibration Using RANSAC Registration," in *Advances in visual computing: 8th Interna-*

tional Symposium, Las Vegas, NV, USA, Dec 2008.

[23] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, "Real-Time Plane Segmentation Using RGB-D Cameras," *Robot Soccer World Cup XV*, vol. 7416, 2012. pp. 306-317.

[24] J. Poppinga, N. Vaskevicius, A. Birk and K. Pathak, "Fast Plane Detection and Polygonalization in Noisy 3D Range Images," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2008. pp. 3378-3383.

[25] S. Yeon, C. Jun, H. Choi, J. Kang, Y. Yun and N. L. Doh, "Robust-PCA-based hierarchical plane extraction for application to geometric 3D indoor mapping," *Industrial Robot: An International Journal*, vol. 41, no.2, 2012. pp. 203-212.

[26] K. Pathak, V. Vaskevicius and A. Birk, "Uncertainty analysis for optimum plane extraction from noisy 3D range-sensor point-clouds," *Intelligent Service Robotics*, vol. 3, no.1, 2010. pp. 37-48.

[27] P. Kohlhepp, G. Bretthauer, M. Walther, and R. Dillmann, "Using orthogonal surface directions for autonomous 3d-exploration of indoor environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 3086-3092.

[28] P. Kohlhepp, P. Pozzo, M. Walther, and R. Dillmann, "Sequential 3D-SLAM for mobile action planning," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 1, pp. 722-729, 28 Sept.-2 Oct. 2004.

[29] K. Pathak, N. Vaskevicius, and A. Birk, "Revisiting uncertainty analysis for optimum planes extracted from 3D range sensor point-clouds," in *IEEE International Conference on Robotics and Automation*, May 2009. pp. 1631-1636.

[30] Maths-Quaternion Notations-As a quantity similar to axis-angle, http://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/geometric/axisAngle

Abstract

This thesis presents a novel three-dimensional registration algorithm based on plane patches. Most algorithms which are used in registration evaluates rigid transformation between two frames by finding correspondence of features in the images. However, lack of information from features or existence of outliers make correspondence incorrect, which causes inaccurate registration of two images.

To solve these problems, the algorithm introduced in this thesis evaluates the rigid transformation by iteratively selecting random pair of plane patches. Each time rigid transformation is determined, similarity of two frames is evaluated by measuring the mean square distance between them. The rigid transformation which minimizes the mean square distance during iteration is selected to output of the algorithm. Finally, calibration of the translation vector which is processed by optimizing the mean square distance of two frames when the rotation matrix is fixed completes the propsed algorithm.

This algorithm has less execution time and more robustness than the algorithms presented before and performance of the algorithm is verified by simulations and real experiments.

Keywords: Plane patch, Registration, Random pair, Mean square distance

Student Number: 2013-23114